

# INDUS GT

Indus Systems Disk Drive Manual  
for Commodore Computer

ERRATA

Notes, Corrections & Updates

1. The Indus GT Disk Drive has been tested for complete compatibility with all current Commodore 64 hardware and software. In most cases, the GT will also operate with older versions of the Commodore 64; however, if you have any problems or want to know if your Commodore is of the most recent vintage, you can ascertain this information by typing:

PRINT PEEK (65408)

and then pressing the RETURN key. You be presented with a number that is to be interpreted as follows:

0 -- The oldest revision, #1.

170 -- The second revision, #2.

3 -- The third, and as of this date most current revision, #3.

If you have an older revision Commodore and are experiencing any problems, you can receive an updated ROM directly from Commodore by ordering their P/N 901227-03 from Commodore Customer Service, CB654, Westchester, PA 19380 (approximately \$15.00, plus shipping) and your unit will be up-to-date.

2. The GT Albert E. Spreadsheet has been updated with an auto-run capability. To load the program, you merely type:

LOAD"GTCALC",8,1

or you type:

LOAD "\*",8,1

or you type:

GTCALC or \* (if the DOS WEDGE is used).

COPYRIGHT ©1984 by INDUS SYSTEMS INC.

This manual is published and copyrighted by Indus Systems Inc. All rights are reserved. This document may not, in whole or part, be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine readable form without prior written consent of Indus Systems Inc.

The word Commodore and the Commodore logo are registered trademarks of **Commodore Business Machines**. **Commodore Business Machines** was not in any way involved in the writing or other preparation of this manual, nor were the facts presented here reviewed for accuracy by that company. Use of the term **Commodore** should not be construed to represent any endorsement, official or otherwise, by **Commodore Business Machines, Inc.**

**Indus Systems Disk Drive Manual  
for Commodore Computers**

By Keith Burgoyne

Copyright (c) 1984 Indus Systems

#### TABLE OF CONTENTS

|  |    |
|--|----|
| Introduction.....                            | 3  |
| Help at the Starting Line.....               | 4  |
| Installation of Drives.....                  | 23 |
| Disk Drive Specifications.....               | 26 |
| Disk Drive Operation.....                    | 27 |
| Introduction to DOS.....                     | 30 |
| Naming Files.....                            | 31 |
| Wild Cards.....                              | 32 |
| Disk Organization and File Types.....        | 33 |
| Disk Drive Extensions of BASIC Commands..... | 34 |
| Disk Drive Commands.....                     | 40 |
| Advanced Disk Commands.....                  | 50 |
| Using the GT "ROM Drive".....                | 53 |
| Fast I/O.....                                | 55 |
| DOS Wedge.....                               | 56 |
| Fast Copy.....                               | 59 |
| Using the GT Utility Diskette.....           | 60 |
| Drive Error Codes.....                       | 70 |
| Care and Use of Diskettes.....               | 74 |
| Noise Radiation Notice.....                  | 76 |

#### INTRODUCTION

This manual will cover two primary functions. First, the installation of your new Indus GT Disk Drive, and second, as an introduction to the operation of the Disk Drive with your computer.

The Indus GT Disk drive has been designed to be fully compatible with all other Commodore peripherals, and its basic operation is similar to the standard Commodore 1541 disk drive. Some of the major differences between the Indus GT and Commodore 1541 include: The addition of an operator control panel equipped with a two digit LED display which gives track location, device number, write-protect and error code status information. A built-in "ROM Drive", that includes commonly needed utility programs -- such as a diskette copy, a disk command short-hand program ("the wedge"), and programs which significantly shorten the amount of time required to load and save programs. A built-in smoked plexiglass dust cover that both reduces any mechanism noise emission as well as decreasing the degree of air-borne contaminants which can cause higher frequencies of soft-errors and reduce the life expectancy of your drive and diskettes.

The Indus GT uses a 5-1/4 inch single or double density, soft or hard sectored "floppy diskette" which is capable of storing and retrieving data in a random-access fashion at a speed far superior to cassette tape. The operating system designed into the disk drive stores the information by file name, controls the allocation of space on the diskette, and does a variety of other housekeeping functions for you.

Besides the manual, enclosed in the dual purpose reusable shipping container, you should have received the Indus GT Disk Drive, a shielded Serial Interface Cable, and an AC power module.

Installing the disk drive and using the operating system are quite simple, as you will learn from the following pages.

#### HELP AT THE STARTING LINE

If you're like most of our GT users, this is the first disk drive you've ever used; and therefore you feel like you have absolutely no idea where to start. But, on the contrary, you've already decided to start in exactly the right place... reading this.

As you may have already noticed, the main portion of the Indus GT Disk Drive Manual for Commodore Computers is by no means a tutorial. Rather, its intended primarily as a reference manual while still trying to be as much help as possible to the new user who doesn't even know what to "refer" to within the manual. Because the rest of our drive manual is not exactly a "perfect fit" for the new drive user, we have also included this little tutorial to assist you in getting your Indus GT off the starting line and into the race.

First, if all you want to do right now is run some games or other pre-packaged diskette programs which you've acquired; then you really shouldn't need to follow this tutorial at all. Simply follow the directions which came with your pre-packaged software.

However, if you want to use your Indus GT for storing your own BASIC programs, then you will probably find this tutorial a good place to start. But before we can start to show you how to use your new GT, you'll need to know how to use your Commodore computer. So... if you haven't done so already, go dig out those manuals which Commodore sent along with your computer and start getting familiar with how to use Commodore's BASIC. Once you've reached the point where you can write a "BASIC" program in which it asks you for your name and then displays your name to you on the screen preceded by "HI", then you're only a short step away from mastering your disk drive as well.

If you are already familiar with your computer enough to write the previously mentioned program, then you should have no problem following this tutorial.

But first, you need to get setup in front of your Commodore computer so that you can try things while I explain them to you. So, connect your Indus GT disk drive as your first drive (device number 8), just like it explains in later in this manual (see Installation of Drives). When you get it setup correctly, you should be able to turn your GT on and have the LED display on the front of it respond everytime you turn your computer off and then on again. Also, your GT should display "8" on the LED displays when you press

the GT's "DRIVE TYPE" button. This "8" tells you that the GT is setup as device number 8. In addition, double check that all of the switches on the back of your GT are set the way the manual indicates; this is very important.

If everything is correct, then you're all set to get started. If something isn't the same as I described it, then recheck your GT's setup against the installation of drives instructions later in this manual. If you have problems with your GT apparently not working, you should detach any other peripherals you may have attached to your Commodore (other drives, printers, modems, tape recorders, etc.); leaving only the Commodore computer and the Indus GT connected together, and then try it again. This will greatly reduce the number of "other" things which may be going wrong with your total computer setup.

If your GT works okay after disconnecting everything else, double check the settings on all of the other devices which you had connected to your computer. Then start re-connecting each device, one by one, re-testing your GT after you add each device. If you have a Commodore disk drive, as well as an Indus GT, then you should take special note of the fact that your Commodore drive is probably setup to be device number 8; and only one device number 8 (as well as any other device number) can be connected to your computer at any one time. Your Indus GT can easily be changed to one of the other disk device numbers (9, 10, or 11); but this tutorial will expect it to be device number 8. So, if your problem is that you have a Commodore disk drive (we at Indus have always felt that Commodore disk drives are a problem), then you should simply leave it disconnected from your computer while you're following this tutorial, and set your GT as device 8.

Lesson number one: We (you and I) are going to stop calling your GT a "disk drive", and we're going to start calling it a "disk device". Although there's nothing wrong with calling your GT a "disk drive", we're going to use "disk device" for the time being so that some of the other terms we'll be using later will become clearer to you. Later, when you're completely familiar with the distinction between a "disk device" and a "disk drive", it will be safe for you to switch back to referring to your GT as a "disk drive".

Lesson number two: This entire tutorial is going to be centered around using BASIC statements to get your GT to do various functions. Because we'll be working in BASIC, you can always use the typing correction features which Commodore described to you in their manuals. However, it is better that you

make sure you have a line typed exactly the way I've shown it to you below before you press the "RETURN" key. Going back and editing a line after you've already pressed "RETURN" may not work exactly right in all cases. Also, read ahead a little before typing in the lines I show you. Sometimes I will explain certain things on the line you should type, after I show you the line, which you'll need to know when typing the line. Reading ahead one paragraph will always be enough to clarify what you should be typing.

Okay, here we go... With your GT all ready to go, but without any diskette in it, type the following BASIC statement as a quick way to get your drive to respond to one of your commands (remember to press the "RETURN" key after each statement to tell BASIC you are finished typing it):

```
LOAD "#1",8
```

Your computer should eventually respond with "READY" to indicate that it has finished performing the statement you entered and is ready for your next statement. After "READY" appears, your GT's LED display should not be flashing. A flashing LED display on your GT indicates an error condition, and there is a very good chance that the switches on the back of your GT are not set correctly. If your computer does not respond with "READY" within 10 seconds, then there is also probably something wrong with the way your GT or another device you have connected to your computer is set up. Double check everything.

Given that the GT's LED display is not flashing (no error), then type:

```
LIST
```

and you should get a listing of all the files which are contained on the "read-only memory drive" ("ROM drive") inside your GT disk device. The LOAD statement you used earlier LOADED this listing into your computer's memory. I'll explain the "ROM drive" later. For now, let's tell BASIC to start running a programmer's helper program by typing:

```
LOAD "#1:#",8,1
```

This "helper", called the "Wedge", will let you type some short-hand commands rather than these long BASIC statements you've been typing so far. Once you press "RETURN" after typing the LOAD statement, BASIC will tell you it is

first "SEARCHING" for the program you requested; and then "LOADING" your desired program. In truth, a special program inside your GT is doing most of this work; it just lets BASIC take all the credit. There will be a short pause between BASIC's "LOADING" message and the Wedge's message which tells you its all ready to go. You won't need to "RUN" the Wedge since it automatically "RUNs" itself once BASIC has finished loading it.

Keep in mind that, when you tell BASIC to LOAD the Wedge, BASIC will LOAD it into memory in pretty much the same manner as it LOADs any other program. This means that any other program which may be in memory at the time will be replaced by the new program being LOADED. So, in the future, you should remember to SAVE any program you have in your computer's memory before trying to LOAD the Wedge. (I'll show you the BASIC SAVE statement later.)

Once the Wedge finishes all of its setup needs, BASIC will say "READY" to indicate you can now enter a new statement; which sounds like a perfect excuse to try out some of the Wedge's special commands. How about typing the Wedge command:

```
#$1
```

This command may look like garbage to you, but remember that the Wedge is providing you with a way of reducing the number of keys you have to type while not interfering with normal BASIC statements. The first key, the "at" symbol ("@"), is one of several keys which, when you use it as the first key you type, tells the Wedge that you are typing a command to which the Wedge should respond, rather than a statement to which BASIC should respond. If you do not begin your line with one of these special keys (I'll explain the others later), then the Wedge will ignore the line you entered; and let BASIC see if the line makes any sense as a BASIC statement.

The "@" key commands the Wedge to take whatever you type following the "@" key and send it to the disk device as a command to which that special program inside the disk device should respond. The last two keys you typed, the "dollar sign" symbol ("\$") and the numeric "1" key, in the above line will command that special program inside your disk device to return a directory listing of all the files contained on the "ROM drive" inside your GT. The Wedge will automatically take the directory listing which that special program sends back to your computer and display it for you on the screen. This saves you from having to separately "LIST" the directory like you did before.

In addition, the Wedge will not "LOAD" the directory listing into your computer's memory as if it were a BASIC program. This means, unlike when you told BASIC to LOAD and LIST the directory before, any BASIC program you may have written will still be in the computer's memory after the directory listing is displayed. This is possibly the greatest feature of the Wedge, since it lets you get directory listings without having to first SAVE any program on which you are working.

Like I mentioned, the "\$1" part of the Wedge command which you typed was sent directly to some "special program" inside your GT disk device. This "special program" is called a "Disk Operating System", or "DOS" for short. (To pronounce this funny word, start off with a "D" like in "Dog", follow it with an "ahhh" as in "open your mouth and say 'ahhh'", and then end it the same way a snake would: with an "sss" as in "hisss". My apologies to those of you older than three years of age, but I couldn't think of a better way of explaining it.)

Since the "\$1" is a DOS command all by itself, we need to examine it more closely. The first key ("\$") is the key which tells DOS that you desire a directory listing. The second key ("1") tells DOS from which one of two "drives" which make up a GT disk device is the one from which you desire the directory listing. If you had typed a zero ("0") instead of a one ("1"), the DOS would have tried to give you a directory listing from the real diskette drive instead of the "ROM drive"; but that would have required you inserting a diskette into the real diskette drive, a process which I'm now about to explain.

Indus sent a "utilities" diskette along with your GT which contains some interesting little "extra" programs which I'll not be discussing in this tutorial, however we can take advantage of the fact that this diskette already contains programs and files of which we can get a directory listing. Go dig this diskette out of the box and I'll explain, incase you don't already know, the correct method for inserting a diskette into the GT..

Have the diskette? Good. First, open the dust cover on the GT (if its not already open) by sliding down the latch panel which is in the center of the drive below the edge of the dust cover. This will make inserting the diskette into the drive just a little bit easier.

Second, turn the GT's door handle counter-clockwise to a horizontal position above the GT's diskette slot. If a piece of white cardboard pops out when you open the door, then you've not yet removed the dummy "shipping disk" from your GT. This shipping disk is used to protect your GT's recording head from damage during shipping. There is nothing holding this shipping disk from within the drive, so just gently pull it out. Don't lose this shipping disk, since you'll want to re-insert it should you ever need to re-ship your drive.

Third, hold the Indus utilities diskette by the label and slide it out of the protective envelope. The envelope is protecting the exposed recording surface of the diskette. Be sure to never touch this recording surface, or expose it to any type of dirt or debris. Also, never try to clean a diskette.

Fourth, insert the diskette into the GT with the label up and the exposed recording surface being the first part of the diskette to enter the drive. As you insert the diskette, the GT's motor should start spinning. The GT will start its motor spinning as soon as it senses the insertion of a diskette to insure a more accurate positioning of the diskette when you close the drive door. Use the indentation in the GT's face-plate as a way to push the diskette completely into the drive; you will hear a "click" when you've pushed the diskette far enough.

Fifth, close the drive door by turning the handle clockwise to a vertical position crossing the GT's diskette slot. Be sure to do this right after you've inserted the diskette so that the motor will still be spinning as the door is closed.

Last, close the drive's dust cover to keep unwanted dust out of the GT.

Now, you are ready to get a directory listing of all the files on this utilities diskette using the same method you used to get a listing of all the files on the "ROM drive". Type the Wedge/DOS command:

`$0`

The only difference between this command and the one you used for the directory listing of the contents of the "ROM drive" is the "0" instead of a "1". The "0" tells the DOS inside the GT disk device to produce a directory listing for the contents of the GT disk drive number zero (the real diskette drive).

By now you have probably started recognizing the difference between a "disk device" and a "disk drive". The "disk device" is the entire Indus GT, which contains two "disk drives". The first "disk drive" (drive zero) is the real floppy diskette drive, while the second "disk drive" (drive one) is the "ROM drive".

And now for the explanation of the "ROM drive" I promised you earlier. Indus has pre-recorded several of the most commonly required utility programs directly within the memory of the Indus GT. This means that these programs will always be available to you regardless of whichever diskette you have inserted in the GT. To make these programs available to your Commodore computer, the GT makes this pre-recorded section of memory look like a second disk drive to your Commodore computer. Since Commodore has included as part of your computer's programming the information necessary for your computer to understand the existence of two "disk drives" as part of a single "disk device", there is nothing non-standard about the "ROM drive" being included as part of the GT.

In addition, since the addition of a second "disk device" (such as another Indus GT or a Commodore 1541) would be treated as a second "device" by your computer which, by itself, could contain two "disk drives", the existence of the "ROM drive" inside your Indus GT will not interfere with your adding a second "disk device" to your computer.

The area of memory inside your GT which contains these pre-recorded programs cannot be re-recorded (written to) with new information, and therefore it is referred to as "read-only memory" (or "ROM"). And, since the GT is making this area look like a second disk drive to your Commodore computer, Indus has called this second drive a "ROM drive".

Well, we've really come a long way so far. You know how to get the Wedge loaded, and you also know how to get directory listings from both a diskette and the ROM drive. In addition, I hope you are gaining a fairly good understanding of what you are doing and not just typing things because I've told you to. (If you're just typing things because I've told you to, then you're missing the whole objective of this tutorial.) Oh, yes... If you've reached this point without encountering any problems with your Indus GT (as opposed to my directions or your ability to follow my directions, be that your fault or mine), then you can feel secure that some shipping clerk didn't knock

something loose inside your GT when he was shipping it to your dealer.

Now, its time to do some real operating of your GT. But first, you'll need to get setup for this next step. You'll need a new diskette which doesn't contain any programs or other information which you wish to keep. If you didn't buy a box of new diskettes when you purchased your GT, then you either had terrific will-power, or the salesperson you dealt with missed his/her chance to increase the size of his/her commission check.

If all you have is the diskettes we sent you with your GT, then you'll need to go buy two or three with which you can play since the Indus diskettes all contain programs and files you'll probably want to keep. If this is the first time you're buying diskettes, then you probably aren't exactly sure what type of diskettes you should buy; so I've provided you with some assistance at the end of this tutorial.

Given that you now have a diskette with which you can play, let's continue with some things you can try. You'll need to remove the Indus utilities diskette from the GT (if you still have it in there). First, open the dust cover the same way you did when you inserted the diskette. Second, open the drive door the same way; this will cause the diskette to pop out a little so that you can get a hold of it. Third, remove the diskette from your GT while being careful not to touch the exposed recording surface. Fourth, return the diskette to its protective envelope by inserting the exposed recording surface into the envelope first. Fifth, insert your "play" diskette into the GT and close the door and dust cover the same way as when you inserted the Indus utilities diskette.

That's enough inserting and removing diskettes, now you are setup to try several things with this single "play" diskette inserted. The first thing you'll always need to do before you can use a brand new diskette is to "NEW" it. To do this, type the following Wedge (and DOS) command:

`NEW0:PLAY,00`

All the "0"s in this command are numeric zeroes and not the letter "O". This command will tell the DOS inside your GT to go out and record (write) special markers onto your diskette which will later tell the DOS where to store the programs or other information you requested it to store. (If your GT starts flashing an error code on its LED displays, then keep reading and, in a

minute, I'll explain how you can ask your GT for an error message which makes more sense than flashing numbers.) Brand new "out of the box" diskettes do not have these markers written on them yet, so DOS will not be unable to store information onto these diskettes until you've told DOS to "NEW" each one of the diskettes.

Once again, the "@" key (or character) in this Wedge/DOS command tells the Wedge that it should respond to the command and that its not a statement intended for BASIC. And, just like with the "@\$" Wedge/DOS command, the characters which followed the "@" character will be passed by the Wedge directly to the DOS inside your GT as a command to which the DOS should respond. In the case of all the "@" Wedge/DOS commands, the Wedge never actually pays any attention to whether or not the command you are sending to DOS is valid, it leaves that decision up to the DOS.

If you leave your GT displaying its current track position (by pressing the "TRACK" button on the GT), you will see it count through all the tracks on the diskette as it writes the special markers on each track. When the DOS is finished writing all these special marks, it will then go to track 18 and record the fact that there are no files currently on this diskette. Track 18 is a special track which DOS reserves for keeping track of all the files you have commanded it to record onto the diskette. This track is called the "directory track", or just "directory", and is where DOS will go next time you request a directory listing of this diskette using the "@ \$" Wedge/DOS command.

The "NEW" part of the DOS command you just used is pretty much self explanatory: it's the command word which told DOS what you wanted it to do. The "@" which followed "NEW" is just like the "@" in the "@ \$" DOS directory listing command. It tells DOS which one of the drive's you want "NEW"ed. If you try "NEWing" drive ":1:" (the "ROM drive"), you'll get a "WRITE PROTECT" error since you cannot write to that drive.

The "PLAY" part of the "NEW" command is the name which you get to select for your diskette. Neither DOS nor the Wedge really care what name you choose, however later sections of this drive manual will explain certain limitations on what names are permitted. The last part of the command was ",00". The "," simply separated the "00" from the diskette's new name, and "00" is a diskette identification code which DOS will write onto the diskette as part of all the special markers. DOS will later use this identification code to make sure you

didn't change diskettes on it at a time when you shouldn't have. You are free to use any characters instead of the "00" in order to create your own diskette identification codes. Needless to say, the only way DOS's diskette change detection safety feature will work is if you give your diskettes different identification codes.

There is another Wedge command which is even more useful (at times) than the Wedge/DOS directory listing command. This command allows you to inquire as to whether or not your GT encountered any errors as a result of the last command you requested it to perform. To see your GT's current error status, simply type:

@

Once again, the "@" character triggers the Wedge to respond to the line you typed and not BASIC. Since the Wedge sees that you have not specified any command text to be sent to the DOS inside your GT, it will ask your GT's DOS for its current error status. After your GT's DOS sends back the error status, the Wedge will automatically display the status message for you. Requesting error status from your drive device using the Wedge has no effect on any program you may have currently in your computer's memory. In fact, with the exception of loading new programs, none of the Wedge/DOS commands will have any effect on any program you currently have inside your computer's memory.

Normally, you don't need to use the error status display Wedge command unless your GT starts flashing an error code on its LED displays. If the GT isn't flashing an error code, then all you'll ever see for a status display will be "OK" since there is no current error for the GT to report. The GT clears its error status back to "OK" right after it is requested to report the status to your computer (either by a request from the Wedge or from a program you write), or when it receives any new command from your computer.

There is one other interesting piece of information your GT will return to your computer which you can then display. But first, you'll need to use a special command to tell your GT to reset itself since you can only get this piece of information back from the GT right after it has reset itself. To tell the GT to reset, use the Wedge/DOS command:

@UJ

The "UJ" DOS command tells DOS to perform a "U"ser jump of type "J". This may sound a little confusing, but all you need to know is that this command causes your Indus GT to reset itself just as if it had just been turned on. When the GT is first reset, it will pretend to have encountered an error of type "73". This is the DOS version error, which Commodore compatible drives indicate when they are reset so that you, the user, can find out which version of DOS is inside your drive. It is now very easy to determine your drive's DOS version number by simply requesting a status display using the Wedge command:

8

With the exception of being able to save any BASIC programs you write onto the diskette (which is probably one of the main reasons you purchased your GT in the first place), and also being able to load your programs back into your computer later (which is the only reason you would want to save them in the first place); you already know all of the day-to-day commands you'll be using to control your GT. So... let's take care of those last two loose ends: LOADING and SAVEing BASIC programs.

BASIC provides two statements for loading and saving programs: LOAD and SAVE (creative, right?). If you've already been saving your programs onto cassette tape, then you are familiar with these statements. These statements also work when it comes to LOADING and SAVEing programs from/to diskettes, except there is an additional ",8" which must be added onto the statement which tells BASIC to SAVE or LOAD your program to/from device 8 (the disk device) rather than device 1 (which is the cassette recorder). In fact, you've already used the disk device version of a LOAD statement earlier (with the addition of ",1" to the statement, which I'll not explain at all in this tutorial) when you loaded the Wedge into your computer: LOAD "1:","8,1".

Before you can LOAD a BASIC program from your "PLAY" diskette, you'll need to SAVE a BASIC program onto the diskette. And before you can SAVE a BASIC program, you'll need to write one. So to start with, type the BASIC statement:

NEW

This will make sure BASIC knows you want to start typing a new BASIC program. And then type this BASIC program into your Commodore:

```
10 GOTO 30
20 PRINT "PLEASE ENTER A NUMBER BETWEEN 1 AND 10."
30 INPUT "WHAT IS THE RADIUS (1-10)";R
40 R=INT(R)
50 IF R<1 THEN 20
60 IF R>10 THEN 20
70 FOR Y=1 TO 25
80   PRINT
90 NEXT Y
100 FOR Y=-10 TO +10
110   FOR X=-10 TO +10
120     IF INT(SQR(X*X+Y*Y))=R THEN 150
130     PRINT ".";
140   GOTO 160
150   PRINT "";
160   NEXT X
170   PRINT
180 NEXT Y
190 PRINT
200 END
```

And then SAVE it to diskette using the Wedge command:

left-arrow: CIRCLE

The "left-arrow" character I referred to is supposed to represent a single key on your Commodore's keyboard: the left pointing arrow. This is not the left arrow key you use to correct typing mistakes (if you happen to make any); its the left arrow key which displays a left arrow on your screen. This single command will cause the Wedge to issue all the commands necessary to your GT in order to save your program onto the diskette in disk drive zero ("0:"). If you had specified ":" instead of "0:", your GT would have complained about a "WRITE PROTECT" error since you cannot save (write) programs to the read-only "ROM drive".

Due to the importance of knowing whether or not the Wedge and DOS were able to successfully save your program, the Wedge automatically performs the "8" Wedge command after the save process has completed. If your program was saved successfully, then an "OK" status message will be displayed. If some error was encountered while your program was being saved, then the appropriate error

message will be displayed.

Given that CIRCLE was saved successfully, type the BASIC statement:

LIST

This will show you that the Wedge did not interfere with the copy of "CIRCLE" which was in memory while it was saving it to the diskette. If you type the Wedge/DOS command:

E\$0

The Wedge will display the directory listing of the files on drive zero, which should only include the CIRCLE program which you just asked the Wedge to save. Following the directory listing, type the BASIC statement:

LIST

This will show you that the directory listing did not interfere with your BASIC program which is currently in your computer's memory.

The DOS inside your GT contains a special protection to prevent you from accidentally selecting the name of an old program already on your diskette as the name for a new program you are trying to save. This prevents you from accidentally replacing the old program with the new one. To demonstrate this, type the same Wedge save command again:

left-arrow0:CIRCLE

This time, you will receive a "FILE EXISTS" error message. This tells you that you already have a file on the diskette with the name you specified ("CIRCLE"), and DOS will not save it using that name. Since you obviously will want to be able to load a BASIC program into your computer, make changes to it, and then re-save it under the same name as you used before, DOS has provided you with a alternate method for doing just that. Instead of telling the DOS to "save" the program, you need to tell it that you wish to "replace" the existing program on the diskette with the new one in the computer's memory. To do this, type the Wedge command:

left-arrow#0:CIRCLE

Once again, I used the "left-arrow" characters to represent the single left arrow key on your Commodore's keyboard. By preceding the file name with the "#at" ("#") symbol, you are telling the Wedge and DOS that you wish them to replace an existing file on the diskette rather than saving a new file. Just as in the case of the save Wedge command, the Wedge will display the DOS status once the file has been replaced. (Special note: Rumor has it that some of Commodore's disk devices do not perform this replace function correctly, so you should be careful when using a disk device from Commodore.)

Terrific, so now you know how to save your BASIC programs onto a diskette (don't forget, you need to "NEW" all new diskettes before you can save programs onto them); but how do you get the programs back into the computer? Okay, first type the BASIC statement:

NEW

This will tell BASIC to erase any existing BASIC program out of the computer's memory. Once you've done that, type the BASIC statement:

LIST

This will let you see that the copy of CIRCLE which was in your computer's memory is gone. Now its time to bring CIRCLE back into your computer's memory, so type the Wedge command:

/0:CIRCLE

This tells the Wedge to start working with DOS to bring CIRCLE back into your computer's memory from the diskette. Once CIRCLE has been completely loaded into your computer's memory, BASIC will display the usual "READY" message. To make sure that DOS did not encounter an error while trying to load CIRCLE back into your computer, check the LED displays on the front of your GT. If they are flashing, then DOS encountered an error and you should use the "#" Wedge command to find out what was the problem. If the display is not flashing, then CIRCLE was successfully loaded.

The Wedge also provides you with a simple command to both load and then run a BASIC program all in one step. To see how this works, type the BASIC statement:

## NEW

Then type the BASIC statement:

## LIST

You will see that the copy of CIRCLE you just loaded into memory has been removed (again). Now type the Wedge command:

"up arrow"0:CIRCLE

This time I've had to use ""up arrow"" to indicate the single key on your Commodore's keyboard which causes an upward pointing arrow to display on the screen. This is not the same key you use to make typing corrections. The Wedge "load and run" command works just like the load command, except the BASIC program is automatically run once it has been loaded into your Commodore's memory.

There are many other things you can do with the Indus GT, which are all covered later in this drive manual; but for now, you know everything you need to know to be able to save and load BASIC programs to/from your new Indus GT for Commodore.

## Buying diskettes is like buying tires:

There are all kinds of various brands of diskettes available in stores today, just like there are all kinds of various brands of tires available for cars. And, just like tires, many of these diskettes will "work" in your disk drive (at varying levels of reliability), and many will not. Just like you can get "steel belted", "radial", and "snow" tires; you can also get "single density", "double density", and "double sided" diskettes. So, you should try to be "diskette smart" before you go out looking for a new box of diskettes.

First thing you should know is that the terms "diskette" and "disk" (sometimes spelled "disc") are often interchanged by salespeople when you are shopping for diskettes/disks. "Diskettes" normally refer to "disks" which are soft and easily bendable (although you shouldn't bend them), like the kind you'll need for your GT. So, "disk" is a general term which includes "diskette".

There are several ways in which diskettes are classified: size, "density", "tracks per inch", and "number of sides". The first classification, size, is the only real classification which makes a diskettes either "work" or "not work" in your GT, but has little to do with how reliable the diskettes will be. The correct size diskette for your Indus GT is "five and a quarter" (5-1/4) inches. These diskettes are commonly referred to as just "five and a quarter disks" for short. If you get a disk which is too large (such as an eight inch diskette), then it won't fit inside your GT. And if you get a disk which is too small (such as a three and a half inch disk), then you won't be able to insert it correctly and the GT won't be able to hold it or record on it correctly. 5-1/4 inch diskettes are by far the most common type of diskette currently available, but this also means you'll have to choose from an enormous number of different types.

Most diskette manufacturers offer various different kinds of 5-1/4 inch diskettes, which are all classified differently using the last three different classifications I mentioned earlier: "density", "tracks per inch", and "number of sides". Let's take the easiest classification first: "number of sides".

If you carefully examine the diskettes which Indus sent along with your GT, being careful not to touch the exposed recording surface, you'll see that both sides of the diskettes have an oval opening which exposes the recording surface of the diskette. It is through this opening that the GT is able to record information onto the diskette. All 5-1/4 inch diskettes have these oval openings on both sides regardless of whether the diskette is "single sided" or "double sided". The "second side" of a 5-1/4 inch diskette is normally the side which also has the label (the "top" side), making the first side (the primary recording side) the side opposite the label (the bottom side).

The difference between "single sided" and "double sided" diskettes is that "double sided" diskettes have been tested by the diskette manufacturer on both sides, and both sides have been certified by the manufacturer as being "good". "Single sided" diskettes have either been tested on only one side, or both sides of the diskettes were tested and the second side was found to be bad. Normally, since it is easier for single sided diskettes to successfully pass testing than it is for double sided diskettes (only half as much of the diskette must be good), single sided diskettes will normally cost less. Which is a good thing for you since your GT only requires "single sided" diskettes.

"Double sided" diskettes will also work with the GT (although they tend to be more expensive), but the GT will only use one side of the diskette.

There are a few 5-1/4 inch diskettes available in stores which are sold as "flippy diskettes". These diskettes get their name from the fact that they are all setup for you to be able to "flip" them over (in single sided drives like the GT) and use the other side of the diskette. This is, of course, a terrific way of recording more information onto each diskette you buy. But! This is absolutely not recommended by Indus or any other serious drive manufacturer. The square jacket, in which the recording surface of the diskette is protected, has a special "brush like" material on the inside which constantly cleans dust and other particles off of the diskette's surface as the diskette rotates. When you "flip" a diskette over, the diskette rotates inside its jacket in the opposite direction; and all the dust which was trapped by the diskette jacket is released back onto the diskette surface.

In addition, single sided disk drives (like the GT) have a special felt pressure pad which gently glides along the opposite side (second side) of the diskette from the side on which the recording head glides (first side). Since no information is supposed to be recorded on the second side of the diskette when using a single sided drive, the felt pad was never designed to protect any information which may be there. When you "flip" a diskette, you will be exposing the first side of the diskette (which contains information) to this felt pad. In most cases, no harm comes from this; but single sided drives are not designed with "flippy diskettes" in mind.

The next classification diskettes are given is "density". All information is recorded onto diskettes in circles (called "tracks") which start at the outside of the diskette (track one in the case of Commodore compatible drives) and continue to the center of the diskette (track 35). The amount of information which is recorded on each one of these tracks is referred to as "density". Some diskettes, when manufactured, are only able to record and retain (reliably) information in "single density"; while other diskettes are capable of recording and retaining information in "double density". "Single density" recording is often referred to as "frequency modulated" (or "FM") recording, while "double density" recording is often referred to as "modified frequency modulated" (or "MFM") recording. There is also another type of recording which is called "group coded recording" (or "GCR"), but diskettes are not normally classified as "GCR" since either "single density" ("FM") or "double density" ("MFM") diskettes will work with "GCR" recording.

Explaining what all these terms mean is a little beyond this tutorial, and you really don't need to understand them to use your GT anyways. All you really need to know is that your GT for Commodore uses "group coded recording" ("GCR"), and the type of "GCR" recording which the GT uses requires a diskette which is classified as "double density". As with single/double sided diskettes, many manufacturers will first test their diskettes as double density diskettes; and then re-test those which failed to see if they will work as single density diskettes.

The last diskette classification is "tracks per inch" (or "TPI"). This term refers to how close together the disk drive records each track of information. Obviously, if tracks are located closer together, then more tracks can be placed on a diskette, and thus more information can be recorded on the diskette. However, for tracks to be located closer together, each individual track must be narrower; and therefore the information which is being recorded as part of each track must be recorded using less of the diskette's surface. Since less of the diskette's surface is being used for each piece of information, those smaller diskette areas must more accurately retain the information.

There are two standard "tracks per inch" being used for 5-1/4 diskettes. The first one is 48 TPI and the second is 96 TPI. The first one, 48 TPI, is more reliable because more of the diskette's surface is used to record each piece of information. The second one, 96 TPI, is slightly less reliable but allows more information to be recorded on the diskette. In addition, drives capable of recording 96 TPI require more expensive components and therefore cost more to build. Also, it is harder to produce diskettes which are capable of retaining information recorded using 96 TPI; and therefore diskettes of this type are also more expensive.

Due to the costs involved, the disk drive Commodore sells for use with their home computers is a 48 TPI drive. And, since diskettes which are recorded using a 96 TPI drive cannot be read by a 48 TPI drive, Indus designed your GT as a 48 TPI drive so that diskettes which you record can be read by another person using a Commodore disk drive. So, since your GT is a 48 TPI drive, all you need to buy is the less expensive 48 TPI classified diskettes. As with the previous two classifications, many manufacturers will first test their diskettes as 96 TPI diskettes, and then re-test those which failed to see if they will work as 48 TPI diskettes.

Because diskette manufacturers first test their diskettes at a higher rating to see if they can be sold at the higher price, it is unlikely you will have much success trying to purchase lower grade diskettes and hoping they will be of higher grade quality. An important point for you to consider is that lower grade diskettes may appear to work, but they may not retain their information over the days, months, or years which you may leave them sitting on the shelf.

Oh, yes... There is a new term with which diskette manufacturer's are starting to play around: "Quad density" diskettes. This term is easily translated into "double density 96 tracks per inch" diskettes. "Quad density" comes from the fact that information is recorded in "double density" on each track, and there are double the number of tracks per inch (96 TPI is twice as dense as 48 TPI). So you'll now know what "quad density" is when a salesperson throws it at you.

In general, buying diskettes is like buying anything else: the best stuff is (usually) more expensive, and you only have so much money you want to spend. Therefore, I offer you a little rule to work from: buy diskettes which are equivalent in quality to the information which you are going to be recording on them. And, dedicate two diskettes for everything you really can't afford to lose. Record a copy of your important programs or files on both diskettes so that if, like it has happened to myself and everyone else, something happens to the first copy, you can always restore the first copy using your second copy. I have completely lost many things on which I have worked for which I had had only one copy, but I have never lost anything for which I've had a second copy. Play it smart!

Okay... in summary, here's the type of diskette you'll want to buy:

5-1/4 Inch  
Double Density  
48 Tracks per Inch (96 TPI/quad is okay but not necessary)  
Single Sided (double sided is okay but not necessary)

Also... there is a section later in this GT drive manual which tells you about the proper method to care for your diskettes.

So... now you know enough to keep the salesperson's commission check to a minimum, and he/she won't be able to overwhelm you with a lot of "diskette language".

## INSTALLATION OF DRIVES

Unpack your new Indus GT Disk Drive along with the accessories included, making sure you save any packing materials for possible future use.

### INSTALLING YOUR FIRST DISK DRIVE

If you are installing the first disk drive on your Commodore system, then carefully follow these step-by-step instructions:

1. Turn the POWER OFF to ALL components of your computer system.
2. Plug the AC power module into the wall socket. Make sure that the POWER switch on your drive is turned off. Now insert the small plug from the power module into the back of the disk drive in the socket marked POWER IN which is located next to the power on/off switch.
3. Plug one end of the shielded Serial Interface Cable into either of the two 6-pin DIN jacks located on the rear of the drive labeled:

"TO COMPUTER, PREVIOUS PERIPHERAL, OR NEXT PERIPHERAL"

Plug the other end of the cable into the SERIAL BUS connector in the back of your computer. Please note: The Indus GT is supplied with a shielded serial interface cable. The use of a cable other than that supplied may defeat the RFI shielding of the communications between the disk drive and the computer and is therefore not recommended.

The two 8 position DIN jacks located at the back of the drive labeled:

"AUXILIARY (see manual)"

have been reserved for future use.

4. Note the group of small slide or rocker type switches numbered 1 through 4 on the back panel. The first two switch positions are used to set the drive device number (08 through 11) and normally come factory set to the Device #8 (Drive 1) positions, which are both on, as illustrated with arrows on the back panel. The "ON" position for a slide type switch is

up, as indicated by an up arrow. The "ON" position for a rocker type switch is the top pressed down. Switches 3 and 4 have been reserved for future use and should be left in their factory set up or ON positions.

5. Locate the power switch at the rear of the disk drive and switch it on. Open the plexiglass door by pressing downwardly on the square button that is centered directly below the door. Now turn the door latch counter-clockwise to the horizontal (open) position.
6. Opening this latch will cause the cardboard, head vibration protector ("shipping disk") to pop out. This should now be removed and saved for re-use whenever your disk drive is to be moved or re-shipped.
7. Turn the power on all of the other components of your computer system, and last of all turn power on to your computer. Now you are ready to use your new Indus GT. You will notice that when any diskettes are inserted into or removed from your disk drive, the spin motor of your disk drive (the motor which causes the diskettes to spin around in their jackets) will momentarily come on and go off after a few seconds, if no drive commands have been received from the computer. This "powered clamping" feature is designed to improve the centering of your diskette in the drive -- your diskette will be more properly centered if the spindle is turning while the door is being closed. You should make it a point to close the door (rotate the knob clockwise) immediately after inserting any new diskettes.

#### INSTALLING MULTIPLE DRIVES

You can connect up to four disk drives, as well as other devices to your Commodore system. The disk drives and tape units are connected to each other in a "daisy-chain" fashion, using the Serial Interface Cables supplied with each device. A set of switches is located on the rear of your drive to set the Device Number.

1. Turn the power off to your disk drive and all other components of your computer system. Looking at the rear of the drive, note the four slide or rocker type switches labeled 1 through 4.
2. The first two switches, numbered 1 and 2, are used for setting the Device

Number (Drive Number) for each drive on the system. For Device #8 (Drive 1) the switches are both set to their on positions, that is, both slides are up or both rocker switches would be pressed down at the top. For Device #9 (Drive 2), switch 2 should be off and switch 1 on. Device #10 (Drive 3) would be the reverse of this with switch 2 on and switch 1 off. Finally, Device #11 (Drive 4) would have both switches 1 and 2 in their off positions. These Device Number/Drive Number switch positions can be easily determined by examining the legend on the rear of the disk drive. Switch positions 3 and 4 have been reserved for future use and should be left in their factory set on or up positions.

3. Connect the second Serial Interface Cable to the unused 6-pin DIN jack on the rear of the first drive and the other end to either of the 6-pin DIN jacks on the rear of the second drive. Repeat this "daisy-chaining" process for each additional drive or peripheral.
4. Restore power to your disk drives, other peripherals, and lastly to your computer. You can verify the setting of your Indus GT Disk Drive device number settings by pressing the drive type buttons on the front of each of the units. If the device numbers do not show the number you intended, re-check the switch settings in the rear of the drive. The computer will recognize the number displayed by the drive type indicator; you can prevent a lot of frustration by verifying device numbers at this time.

### DISK DRIVE SPECIFICATIONS

HEIGHT X WIDTH X DEPTH (OVERALL): 2.65" x 7.25" x 11.0"  
 CAPACITY: Total 174,848 bytes per diskette; 168,656 bytes per diskette with Sequential files; 167,132 bytes per diskette with Relative Files  
 DIRECTORY ENTRIES: 144 per diskette  
 NUMBER OF TRACKS: 35  
 DATA ENCODING METHOD: Group Coded Recording (GCR)  
 NUMBER OF SECTORS PER TRACK: 21 (tracks 0 to 20); 19 (tracks 18 to 24); 18 (tracks 25 to 30); and 17 (tracks 31 to 35)  
 NUMBER OF HEADS: 1  
 TRACK DENSITY: 48 tracks per inch  
 BYTES PER SECTOR: 256

| ENVIRONMENTAL: |                              |                             |
|----------------|------------------------------|-----------------------------|
|                | Operating                    | Non-Operating               |
| Heat:          | 10 to 50 C<br>50 to 115 F    | -40 to 71 C<br>-40 to 160 F |
| Humidity:      | 20 to 80 %<br>Non-Condensing | 5 to 95 %<br>Non-Condensing |
| Altitude:      | -500 to +50,000 feet         | -500 to +50,000 feet        |

DATA TRANSFER RATE: 5.3 K Baud (nominal)  
 MAXIMUM BIT DENSITY: 5162 flux changes per inch  
 ROTATIONAL SPEED: 300 RPM  
 SOFT ERROR RATE: 1 in 10<sup>9</sup> bits  
 HARD ERROR RATE: 1 in 10<sup>12</sup> bits  
 SEEK ERROR RATE: 1 in 10<sup>6</sup> steps  
 MEDIA LIFE: 3.2 x 10<sup>6</sup> passes per track  
 MTBF: 10,000 Power On Hours  
 MTTR: 30 minutes  
 POWER REQUIREMENTS: 12 VDC, 2.5 Amps (Nominal)  
 NET WEIGHT: 4 lbs. 14 oz.  
 MEDIA REQUIREMENTS: Industry Standard (ANSI) 5-1/4" diskettes, hard or soft sectored

### DISK DRIVE OPERATION

Your new Indus GT Disk Drive has a number of operational features that will be described in this section. Among the most prominent of these features, and the feature we will first describe is the operator control panel located on the front of the drive. Other features and unique characteristics of the GT will then be briefly discussed.

The operator control panel, is located behind the dust cover, and contains a two-digit LED display, three individual LED (Light Emitting Diode) status indicators, and four pushbuttons.

The three LED's in the center are each labeled to represent their functions. The POWER LED is illuminated whenever power is applied to the drive. When the drive is in use (reading or writing) the BUSY LED is illuminated -- letting you know not to insert or remove a diskette. The rightmost LED marked PROTECT is illuminated whenever the diskette in the drive has been write protected, either by covering the square write-enable notch on the diskette, or by having pressed the adjacent button to set the write protect mode electronically. This button is a very useful feature as it allows you to write protect a diskette without having to remove it from the drive to place a sticker over the notch.

The other three pushbuttons to the far right set the display mode for the two digit LED display on the far left. The digital displays will show the current track location of the read/write head as the default unless an error occurs, in which case the error code will be displayed automatically.

Pressing the button labeled DRIVE TYPE will display the device number of the drive (08 through 11) that have been set by the switches on the back panel of the drive. Since the drives can have their device number assignments redefined through software, the digital displays will show the current "logical" device number if the switch settings on the rear panel of the drive have been overridden due to some software command. If you wish to temporarily re-assign the device number of your drive without re-setting the switches on the back of the drive you can do so by holding down the DRIVE TYPE button and momentarily depressing the adjacent TRACK button; doing this will allow you to sequence through each potential drive device number (08, then 09, then 10, then 11).

Pressing the TRACK button will switch back to displaying the current track location of the drive's read/write head. It should be noted that some copy protected programs will occasionally by-pass the normal DOS head positioning routines and directly control the stepper motor. These programs will still work with the Indus GT Disk Drive, but may cause the digital displays to discontinue reflecting the true current track location of the drive. These programs will usually revert to using the DOS positioning routines once they have accomplished their "protection" system and the displays will again accurately reflect the current head/track location.

The ERROR button will allow the display of the last error code encountered. If a 00 is displayed then no current error exists on the disk drive. Since there is no track 00 on the drive -- its first track being 1 -- it is easy to distinguish the 00 as an error condition, not a track. All other error conditions will flash, thus distinguishing them from track location data. When encountering an error during the performance of any normal disk command, the appropriate two digit error code shown later in this manual will be flashed on the digital displays. Any subsequent valid disk command received by the drive will cause the error message to clear, showing 00 as the current error status. Many copy protected programs will cause varieties of diskette errors in a deliberate attempt to foil unauthorized duplication of their products. The resultant error messages and subsequently cleared 00 indications are no cause for alarm; the GT Disk Drive will perform its normal duties and is merely trying to inform you of peculiar events taking place on your diskette.

When inserting a diskette into your drive, it is necessary to continue pushing the diskette into the drive until you hear and feel it click into place. It will be easier to accomplish this task (especially with long fingernails) if you push the edge of the diskette nearest the drive faceplate finger depression area. The door knob must be rotated clockwise to a vertical position in order to engage the diskette clamp. Opening the door by rotating the knob counter-clockwise, will cause the diskette to be popped partially out, for easy removal.

In order to cause your diskette to be more properly seated and centered with respect to the drive hub and read/write head, Indus has incorporated a "powered clamping system" in the GT Disk Drive. When a diskette is inserted into or removed from the drive, the drive's spin motor will begin rotating for several seconds. Engaging the diskette clamp (by closing the drive door) when

the diskette is spinning will considerably improve the diskette seating and centering accuracy -- you will experience longer diskette life, better data integrity, and fewer frustrations with unsuccessful diskette loading as a result of this precaution.

The GT Disk Drive comes equipped with a built in dust cover. The cover can reduce the frequency of accidental access to your diskettes by uninvited visitors (you won't unintentionally bump the knob or write protect switch while you are saving an important program either!). It will also reduce the amount of airborne contaminants which reach your diskette and drive read/write head surfaces. Over a period of time the cumulative results of less wear and otherwise improved life expectancy and performance of your drive and diskettes can be substantial.

## **INTRODUCTION TO DOS**

The purpose of a disk operating system (DOS) is to provide an easy way for your computer (and you) to communicate with your disk drives, printer, and other peripherals. The disk operating system contains commands and utilities which allow you to:

1. Organize the information and programs with which you are working into "files". This organizing is similar to the way you would organize information printed on paper using file folders and a filing cabinet.
2. Providing an easy means by which you can access this information whenever needed.
3. Make use of various pre-written specialized application programs (like wordprocessors, spread sheets, and spelling checkers) and programming tools (like higher powered BASICs, machine/assembly language processors, and program "debugging" aids).
4. Go from using the disk operating system, to using any of the thousands of diskette based programs, back to using the DOS itself again.

But, in order for the DOS to know exactly what you want it to do at any given moment, you need to "command" it. And "command" is truly the correct word for how you talk (using the Commodore's keyboard) to the disk operating system. Whatever you command the DOS to do, it will do provided it knows how to do it and provided you have commanded it in a way it understands. Remember, the DOS never actually "thinks" for itself (although beginners often find this hard to believe); it only does what you tell it to do regardless of whether or not what you told it is actually what you wanted to do. Explaining exactly how to command the DOS and disk drive to do your bidding is the sole purpose of this manual.

### **Naming Files**

Much like the way a record or cassette tape can hold a number of songs, a single diskette can hold many distinct files of information (up to 144 files per diskette). These files can hold programs or data in text (human readable), binary (computer readable), or several other forms. Just like songs on a record, diskette files must have names so that you can instruct the drive exactly which file you wish to use.

The DOS contained in your GT Disk Drive will only recognize the name of a file stored on a diskette only if the name is written in a particular way. A file name should contain from one to 16 alphanumeric characters. It should nearly always be enclosed in quotation marks. Symbols such as @, \*, ?, and \$ as well as some others should be avoided, as they are special reserved words for certain BASIC and/or DOS functions.

Through the years, computer "hackers" have found it helpful to add a file extension to help identify more easily the types of files that have been written. A list of the more common filename extensions has been provided below :

BAS for a "SAVE'd BASIC program (MYPROG.BAS)

LIS for a "LIST"ed BASIC program (MYPROG.LIS)

COM for a COMMAND/utility program (FAST COPY.COM)

EXC for an EXeCute file (STARTUP.EXC)

SYS for a SYStem file (GTHELP.SYS)

TXT for a wordprocessor or editor file (LETTER.TXT)

In some cases, you will need to specify the drive number (0: or 1:) before the filename and within the quotation marks. This was especially true of the older dual drive Commodore units, but is also true to a certain extent with the GT Disk Drive because of its "ROM Drive" being identified as drive 1:.

Proper DOS communication has the device number following the filename, separated by a comma. In general, however, if you do not use "1:filename", DOS will assume that "filename" means "0:filename".

Whenever you command the DOS to do something which causes it to create a new file on a diskette, the operating system adds the name of that file to something called a "directory". By maintaining this directory, the DOS knows where each file on your diskette is located just like your address book or telephone directory tells you where all your friends and associates are located. Whenever you tell DOS you wish to use one of the files on your diskette, the DOS reads through this directory, entry by entry, until it finds the entry for the file you requested.

#### Wild Cards

To enable you to be more selective and to eliminate excess typing, a special feature has been installed in the DOS called the "wild card". This is similar to the joker in a deck of cards, where it can be substituted for any other card. The same can be true with file names, which enables manipulation of groups of files as we shall see.

There are two types of wild cards allowed:

1. An asterisk (\*) which can stand for any combination of numbers and letters, or
2. The question mark (?) which stands for only one letter or number.

Let's say you have 15 letters on a diskette (thinking ahead you had labeled all of your letters as "LETTER-JIM", "LETTER-MOM", etc.) which also contains 30 miscellaneous other files, and you would like a directory listing of just your letters. Instead of having to look at the entire directory of 45 files, you could enter LOAD "\$:LETTER\*", 8 and then LIST it, instead of just pressing RETURN again and again when obtaining a normal directory listing. This option would just list the 15 letters (with the names beginning LETTER) and not the other miscellaneous files. The same wild card feature will work with most other disk commands.

The question mark (?) is used as a substitute for a single letter or number in a file grouping such as FILE1.TXT, FILE2.TXT, FILE3.TXT, and FILE4.TXT. In this case the question mark could be used in a filespec or filename in place of the number (e.g., FILE?.TXT would match each of these files).

#### Disk Organization and File Types

Your diskette is organized into 35 tracks or concentric circles of data (the tracks are numbered from 1 to 35). Each of these tracks is further broken up into blocks or sectors containing 256 bytes of information each. The number of sectors contained on each track varies as you will note below:

| Track Range | Number of Sectors |
|-------------|-------------------|
| 1 - 17      | 21                |
| 18 - 24     | 19                |
| 25 - 30     | 18                |
| 31 - 35     | 17                |

Although a total of 683 of these sectors or blocks are contained on the diskette, the 19 sectors committed to track 18 for directory or file location purposes, reduces this number to 664 sectors into which data may be placed.

Four fundamental file structure types are supported with your disk drive: PRogram Files, SEquential Files, RElative Files, and USer files. The Program

and Sequential files are the most commonly used file types, and have very common file structures and directory track utilization patterns.

Both program and sequential files place the names of their files, together with their beginning track and sector locations in the directory area of track 16. Each successive sector contains as the first two bytes in each sector a forward link to the next sector contained in this file. When no forward sector pointers are encountered it is assumed that the final sector of the file has been loaded. Enough directory entry space is available to support up to 144 file names and their respective starting track and sector locations.

Relative files directly control the location of related data, which is even permitted to cross sector boundaries. These files are located from track and sector side block directories, never more than one to three sectors away from any data you may need. Whereas a sequential or program file of 86 sectors in length could be as many as 86 sectors away from the information you may need.

USER files are basically a do it yourself file structure. You can create your own filing and linking techniques based on the one of the DOS supported methods or create one of your own construction.

#### DISK DRIVE EXTENSIONS OF BASIC COMMANDS

Many of your normal BASIC commands have additional usages and functions, requiring special syntax or command structure, when used to command a disk drive. These BASIC commands will be dealt with below primarily with regard to their application to disk drives -- see your Commodore user's manual for their normal usages and command structures or syntax.

Command Name: LOAD

Purpose!: This command permits you to retrieve a file you have chosen from the disk device number and disk drive number you have specified.

Syntax1: LOAD"driveno:(optional)filename",deviceno,commandno(opt.)

Example1: LOAD "0:MYPROG.BAS",8  
LOAD "MYGAME.PRG",8,1

Purpose2:

The LOAD routine can request that a special directory function be performed by requesting a "#". This directory, once loaded, can be LISTed to see a list of the files contained on this disk.

Syntax2:

LOAD"\$driveno:(opt.)filename(opt.)",deviceno

Example2:

LOAD "#",8  
LOAD "#1:T\*.BAS",9

Arguments:

LOAD -- a BASIC command that may be used in either direct (immediate) or indirect operating modes to retrieve files from specified input devices.

directory-command -- a special LOAD function invoked by a \$ being placed within quotes as a command pre-fix to the file(s) and devices being accessed.

driveno -- the drive number, 0 or 1, within each device number.

filename -- the name (up to 16 alphanumeric characters and/or combinations of wildcards) of the file you wish to use.

deviceno -- the device number (disks are normally 08 through 11) you wish to use for this command and file.

commandno -- the secondary command specifier. A 0 or no specifier will cause the file to be loaded at the beginning of BASIC's workspace (normally 2049 or #0801). A 1 will cause the program to be loaded to the position indicated on the sector's bytes 4-3 -- many pre-packaged programs will begin automatically executing when this choice has been specified.

Unlike cassette usage, LOADING from a disk requires that specific filenames and device numbers be identified for the function to be performed. Instead of instructing you to PRESS PLAY on your cassette recorder, you will be informed

that BASIC is SEARCHING FOR filename, then that it is LOADING. The LOAD function when executed will interact with the Disk Operating System within your disk drive to first verify that the device number you specified is available, whether the filename you requested exists on this drive, locates the beginning track and sector location of your file if it does exist, and continues loading all of the related sectors or blocks of information contained in your file on the various tracks and sectors on which they are located until the last block is LOADED into your computer's memory, at which time you will be notified that the LOAD function has been completed by BASIC printing it's READY prompt on your screen.

**Problem Preventor:** A LOADED and/or LISTed directory command (\$), remains in the computer's memory and should be eliminated (by typing NEW) before LOADING and RUNning any program to prevent any potential conflict the directory listing might present. Also, a directory loads into memory just like a BASIC program and therefore it will replace any BASIC program you already have in memory. Be sure to SAVE any BASIC programs before loading a directory (see "DOS Wedge" for a solution to this problem).

As has been previously noted, since the use of drive number 1: has been invoked in your Indus GT Disk Drive, it will pay to specify the normally optional drive number parameter more frequently than you might otherwise have done. This can never hurt, and will often save you some problems even when using Commodore's 1541 disk drive with certain programs, etc.

**Command Name:** SAVE

**Purpose1:** This command permits you to save a program currently residing in memory to a disk drive, using the file name which you specify

**Syntax1:** SAVE "driveno:(optional)filename", deviceno

**Example1:** SAVE"MYPROG",8  
SAVE "0:LETTER.TXT",9

**Purpose2:** This command has an additional save and replace capability, using the # parameter, which is required since a normal BASIC SAVE will not always let you write over a

previously existing file with the same name.

**Syntax2:** SAVE "replace-command driveno:(opt.)filename", deviceno

**Example2:** SAVE "#C:CHANGEDPROG",8

**Arguments:** SAVE -- a BASIC command that may be used in either the direct (immediate) or indirect mode to transfer programs in the computer's main memory to a tape or disk device.

replace-command -- the # symbol used as a command parameter prior to the driveno or filename arguments to force writing of the filename over a previously existing file with the same name.

driveno -- the drive number, 0 or 1, within each device number.

filename -- the name (up to 16 alphanumeric characters and/or combinations of wildcards) of the file you wish to use.

deviceno -- the device number (disks are normally 08 through 11) you wish to use for this command and file.

The SAVE command is a very useful BASIC command which allows you to transfer programs currently residing in the computer's volatile (subject to erasure with loss of power) RAM memory to a more permanent storage media such as tape or disk, for later recall or usage. The SAVE command when used with disk drives, requires filename and device number specifications that were not required when using cassette tape devices.

When activated the SAVE command will display SAVING filename. It will interact with the Disk Operating System located within your drive to: 1. See if the same filename has been previously used (it will overwrite it if the "#" command has been invoked). 2. It will see if there is enough directory space on the diskette to write your new file's name (maximum number of directory entries is 144). 3. See if enough free blocks or sectors are available on which to write your new file (although a total of 683 blocks or sectors exist on the disk, a maximum of 664 are normally available for program storage). 4.

Lastly, it stores your program on the diskette and updates the Block Availability Map of your program's sector usage and filename, etc. Any number of errors can be encountered during this process that will cause your disk drive lights to flash an error code corresponding to those outlined in a later section of this manual.

When the SAVE command has completed it's task, you will be so informed through it's printing of a READY message on your screen. Although your program has been saved to disk, it still exists in your computer's memory. You can verify this by typing LIST to see your program on the screen.

**Problem Preventor:** It is a wise practice to periodically save long programs on which you are working. Using the SAVE and REPLACE function can be invaluable at these times. It can also pay to occasionally save alternate names for your program as a sort of back-up or archival trail of your development efforts -- you won't have to start from scratch if catastrophe strikes!

**Problem Preventor:** Although it is a rare occurrence, using the VERIFY command at this point could prove to be a wise investment of time. Since your program is already in memory, it will be very easy to have the RAM image of your program compared with that stored on disk to make certain no glitches or errors have occurred. The disk drive has a very sophisticated method of ensuring these errors are prevented and/or detected during the save process; your concern with the validity of your data should be proportionate to its importance to you.

**Problem Preventor:** Always be sure the drive is not flashing an error number after a SAVE, SAVE & REPLACE, or VERIFY!

Again, although it is generally more common to specify a drivenumber parameter when using SAVE commands, accentuating it's usage for the GT Disk Drive cannot hurt. Since protected programs often utilize undocumented elements of the operating system, it is difficult to predict when or where they might fail to return to the correct drive number when leaving or exiting from their normal functions.

Command Name: VERIFY

Purpose: Used to compare a program in memory with one stored on

disk or tape.

Syntax: VERIFY "driveno:(optional)filename", deviceno

Example: VERIFY "MYPROG",8  
VERIFY "0:NEWPROG",9

Arguments: VERIFY -- a BASIC command accessible from both the direct (immediate) and indirect modes, used to compare for exact equivalency of data in memory with the specified file on tape or disk.

driveno -- the drive number, 0 or 1, within each device number.

filename -- the name (up to 16 alphanumeric characters and/or combinations of wildcards) of the file you wish to use.

deviceno -- the device number (disks are normally 08 through 11) you wish to use for this command and file.

The VERIFY command is an invaluable BASIC command used to compare byte for byte between a RAM resident program and one located on disk or tape. The use of VERIFY with a disk drive requires the specification of filename and device number parameters.

**Problem Preventor:** A program SAVED from a VIC-20 (having a normal BASIC load address of 4096 (or \$1000) will not compare byte for byte (since each line of BASIC code contains a pointer to the absolute memory location of its next line number) with one you have loaded into your Commodore 64's normal basic load area of 2049 (or \$0801). You can save yourself some unnecessary concern by avoiding guaranteed problems of this nature.

When the VERIFY command has been activated you will be informed that it is SEARCHING FOR filename, then that it is VERIFYING. If successfully verified, you will be so informed by having OK and then READY displayed on your screen. An unsuccessful verify will cause you to be greeted with a ?VERIFY ERROR on your screen, and a flashing light on your disk drive.

## DISK DRIVE COMMANDS

A description of the most commonly used disk drive commands, their syntax or way of expressing these commands, and a review of the functions performed by these commands will follow. You should also review the chart of Wedge/DOS comparisons contained in the next section of this manual.

These commands generally communicate with the disk drive through use of the computer's command channel (Channel 15). The format of the following disk commands typically makes use of the OPEN statement to open the logical file#, the device#, the channel#, and optionally the command\$ (command string) or text\$; then to use the ?PRINT# statement to force the command itself on the command channel; and then finally to close the logical file#. A typical command will take the form:

```
OPEN 15, 8, 15
PRINT#15, "NEW0:diskname,id"
CLOSE 15
```

Where:

The logical File# is the number you use to tell BASIC which OPEN, PRINT#, INPUT#, and CLOSE statements go together. Only you, your programs and BASIC pay any attention to file#'s, the device you talk to never sees the file# you used.

Device# is the number you use to tell BASIC the device to which you wish to talk whenever you specify a PRINT# or INPUT# with the same File# as the one you used when OPENing the File#. A list of these device numbers follows:

- 0 = Keyboard
- 1 = Cassette Recorder
- 2 = RS-232/User Port
- 3 = Screen
- 4,5 = Printer
- 8 - 11 = Disk

The Channel# is the means by which the device provides you with the capability of having more than one file open for reading and writing on a single device at one time. Without Channel#, the device would not know which one of several files you have

open is the one from which you wish to read or write. For disk drives, some Channel#s have reserved meanings:

Channels 0 & 1 are reserved for DOS's own use and you should not use these Channel#s.

Channels 2 - 14 are available for you to use for reading from and writing to your data files from within BASIC programs.

Channel 15 is reserved as the means for you to send special commands to your disk device (such as "NEW" a diskette, or "SCRATCH" a file), which the disk device should not get confused with data to be written to a file. Only you, your program, and the device pay attention to the Channel#. BASIC simply passes the Channel# to the device for you, without paying any attention to its value.

The PRINT# command is not the same as the PRINT command in BASIC. Although it performs essentially the same function as BASIC's PRINT, it has a special form to associate it with file output. It cannot be abbreviated with a ?# nor should you put a space between the PRINT and the # symbol. The PRINT# statement is followed by a number that refers to a device or file number that was previously opened. The file or device number is then followed by a comma, followed by the command or data to be printed.

The INPUT# (reading to the next RETURN) and GET# (reading a single byte) statements, also used specially on the command and communication channels, are similar to the PRINT#, in their functional equivalency to their BASIC counterparts, in their referencing a previously opened file or device number and in their being undetectable from their # symbol.

NOTE: PRINT# and INPUT# use File#s and not Channel#s. But Channel# 15 is the command channel for disk devices, therefore:

```
OPEN 6, 8, 15
```

```
PRINT#6, "NEW0:INDUS,00"
CLOSE 6
```

works just as well as:

```
OPEN 15, 8, 15
PRINT#15, "NEW0:INDUS,00"
CLOSE 15
```

in fact, you can even do:

```
OPEN 1, 8, 15
OPEN 2, 9, 15
OPEN 3, 10, 15
OPEN 4, 11, 15
PRINT#1, "NEW0:INDUS8,00"
PRINT#2, "NEW0:INDUS9,00"
PRINT#3, "NEW0:INDUS10,00"
PRINT#4, "NEW0:INDUS11,00"
CLOSE 1
CLOSE 2
CLOSE 3
CLOSE 4
```

File# 15 is often used in examples to associate it with the command channel 15, but this practice can also be somewhat confusing.

The commonly used disk commands outlined below, deal with the primary disk housekeeping functions you will need on a day-to-day basis. Advanced commands, allowing you to load and execute code within the disk drive directly, will be presented in outline form at the end of this section, but are generally felt to be beyond the scope of this document to explain at any length.

Command Name: NEW

Abbreviation: N

Purpose: Prepares a new, blank diskette to accept data. Builds tracks, sectors and directory/block availability data. Names and otherwise identifies the diskette.

Syntax: PRINT# fileno, "NEW driveno:(opt.)diskname,diskid(optional)"

Example: PRINT# 15, "NO:MYDISK,01"
PRINT#15, "NEW:MINEDISK"

Arguments: PRINT# -- prints the following command to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been opened using the disk device's command channel number (15).

driveno -- the drive number, 0 or 1, within each device number.

diskname -- a name of up to 16 alphanumeric characters by which you wish to identify your diskette. This name will appear whenever a directory listing is made of your diskette, and is otherwise purely cosmetic and will not affect your files in any way.

diskid -- this is a two byte identifier that is subsequently written to every block or sector on the diskette and is used by the Disk Operating System contained in your drive to see if you have switched diskettes without letting it know. If each of your diskettes contains a unique identifier, it will be less likely for the DOS to inadvertently write on or alter one of your diskettes by mistake. The diskid cannot be altered by any normal DOS command other than another NEW command. If the diskid number is omitted, the drive will erase and rewrite the directory and block availability information contained on track 16 sector 00 and following, but will not re-format the tracks

and sectors on the diskette.

**Problem Preventor:** A diskette that has the NEW command performed on it (i.e., is formatted) will no longer contain any previously written data on it. Please be certain you know which diskettes you are formatting, and the value of any programs that might be on it. They will be irretrievably lost when this process has been completed.

The NEW command clears a diskette of any previous data, and prepares it for your future use -- similar to the way BASIC's NEW command clears the computer's memory so that any new programs you wish to use will not become confused with the old ones or find they have inadequate RAM memory in which to operate. The NEW command is one all Commodore users, experienced and novice alike, get to perform on a regular basis. Blank diskettes are generally not provided with any computer usable formats on them, although they may have been checked for their magnetic characteristic integrity. Indus GT Disk Drive users will find that the NEW command takes only a fraction of the time required by a 1541 owner, even when the FAST I/O routines of its ROM Drive have not been engaged.

Command Name: INITIALIZE

Abbreviation: I

Purpose: Forces the drive to examine the diskid and Block Availability Map of the currently installed diskette. Clears the error channel, and restores the drive to its power on condition.

Syntax: PRINT# fileno, "INITIALIZE driveno(optional)"

Example: PRINT#15, "INITIALIZED"  
PRINT#15, "I"

Arguments: PRINT# -- prints the following command to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been

opened using the disk device's command channel number (15).

driveno -- the drive number, 0 or 1, within each device number.

During its normal power up process, and also when the computer power is turned on, the Disk Operating System resident in the drive will examine the diskid number of the diskette currently installed and will also read the Block Availability Map (how many sectors have been used by previous files and where those sectors are located) and will store this information in its internal memory. This permits the drive to know very quickly if it will have enough room to store your new program, and if you switched diskettes, that it cannot presume that it knows where any used or empty sectors on this new diskette are located. When a new diskette is inserted, the drive and DOS take elaborate precautions to make certain no inadvertent damage is done to the new diskette. A change in diskid numbers will cause the drive to become reluctant to perform certain functions on the new diskette (you may not have realized, as it has, that the diskette is different -- you haven't given your drive the go-ahead or "all's clear" password via an INITIALIZE command).

Your INITIALIZE command lets the drive know that it will be working with a new diskette now -- new diskid and new BAM. It will restore itself to its power on condition, clearing any previous error conditions and flashing lights, and place the new diskette data in its own local memory.

You should get in the habit of sending an INITIALIZE instruction to your drive each time a new diskette is inserted. An ounce of prevention is worth a pound of cure!

Occasionally in a two or more drive system, the second and third drives will need to have an INITIALIZE command sent to them enabling the system to recognize their presence (otherwise DEVICE NOT PRESENT errors).

**Problem Preventor:** An INITIALIZE command will return a 21 error (No Sync Character) if it is asked to examine a non-existent or unformatted diskette. Making certain the drive is capable of executing the command you are giving it will lessen your frustration with continued flashing errors.

Command Name: COPY

Abbreviation: C

Purpose1: This command allows you to retrieve a file from a diskette and store it back to the same drive under a different name.

Syntax1: PRINT# fileno, "COPYdrive:(opt.)newfile=drive:(opt.)oldfile"

Example: PRINT# 15, "C0:MYNEWFILE = 0:MYOLDFILE"  
PRINT# 15, "COPY:MYNEWFILE = MYOLDFILE"

Purpose2: This command may also be used to combine two through four files into a single new file.

Syntax2: PRINT# fileno, "COPY drive:(opt.)newfile = drive:(opt.)oldfile1, drive:(opt.)newfile2, drive:(opt.)newfile3, drive:(opt.)newfile4"

Example2: PRINT#15, "C0:MYNEWFILE=0:MYOLDFILE1,0:MYOLDFILE2,0:MYOLDFILE3"  
PRINT#15, "COPY:MYNEWFILE = MYOLDFILE1, MYOLDFILE2, MYOLDFILE3,  
MYOLDFILE4"

Arguments: PRINT# -- prints the command which follows it to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been opened using the disk device's command channel number (15).

drive -- the drive number, 0 or 1, within each device number.

newfile -- title of the new file name consisting of up to 16 alphanumeric characters that you wish to call your newly copied (or concatenated) file.

oldfile -- title of the file (or files) from which your new disk file name is to be created.

Since the COPY command only permits files to be copied to the same device number, and on the same diskette it is of fairly limited usefulness. Even so, it can be useful for making progressive back-up copies of a working file without having to load any utility diskettes or programs. The concatenation capability (ability to combine several source files) is rarely utilized, but it's nice to have around just in case.

Command Name: RENAME

Abbreviation: R

Purpose: Use of this command permits you to change the name of a file that is already written on your diskette.

Syntax: PRINT#fileno,"RENAMEdrive:(opt.)newname=drive:(opt.)oldname"

Example: PRINT#15, "R0:MYPROG.VER1=0:MYPROG"  
PRINT#15, "RENAME:MYNEWPROG=MYOLDPROG"

Arguments: PRINT# -- prints the following command to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been opened using the disk device's command channel number (15).

drive -- the drive number, 0 or 1, within each device number.

newname -- title of the new file name consisting of up to 16 alphanumeric characters that you wish to call your newly copied (or concatenated) file.

oldname -- title of old file currently residing on your diskette which is to be renamed by this command.

The rename command allows you to change the names or titles of the programs you have previously recorded on your diskette without resorting to sector editors and directly examining, decoding, and resaving the directory

information located on track 18, sectors 00 and following of your diskette. File not found errors, etc., can be reduced, by first creating a directory listing of the exact file name and its spelling prior to renaming it.

Command Name: SCRATCH

Abbreviation: S

Purpose: This command allows you to delete or remove a file from your diskette once you no longer need it.

Syntax: PRINT# fileno, "SCRATCH driveno:(opt.)filename"

Example: PRINT#15, "S0:OLDFILE"  
PRINT#15, "SCRATCH:OLDFILE"

Arguments: PRINT# -- prints the following command to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been opened using the disk device's command channel number (15).

driveno -- the drive number, 0 or 1, within each device number.

oldfile -- title of the old file name consisting of up to 16 alphanumeric characters that you wish to delete or SCRATCH from your directory.

The SCRATCH or delete command allows you to eliminate from your diskette a program that you no longer wish to keep. The command causes the DOS to place a 00 file type next to it's directory entry (DELETED) and will then trace through the old file's track and sector links, de-allocating each sector or block from its being used status, to be available for allocation for any new files. A normal directory request (LOAD "\$", 8 then LIST) will no longer show your old file's name once the SCRATCH command has been performed. The XXX BLOCKS FREE message following the directory listing will now be as many blocks larger as your old file previously occupied.

**Special Note:** Since the SCRATCH command does not directly damage the directory information or the information actually contained in each of it's sectors on the diskette, it is possible under some conditions to restore an inadvertently SCRATCHED or deleted file. Please refer to the section in this manual which discusses the GT Utility Diskette for more information regarding this possibility. However, it is always preferable to avoid scratching wanted files.

**Problem Preventor:** Do not attempt to SCRATCH or delete a file whose file type is preceded with an asterisk (\*) in a directory listing. The asterisk is an indication that the file was not properly closed, and disastrous results can ensue. The last record or block in an unclosed file will contain pointers to the next sector in which it intended to store data. This next sector it is pointing to may have been unallocated at the time, or may contain data from a previously SCRATCHed file (or even subsequently saved file since the sector will remain unallocated in the BAM). In it's process of tracing from sector to sector de-allocating SCRATCHed file blocks, the program would have no way of identifying that the data in the next cell did not really belong to the SCRATCHed file and would trace through the other file's sector pointers de-allocating those sectors as well. Subsequent use of the diskette for almost any purpose would continue to corrupt the diskette. Diskettes containing unclosed files should have as many files as possible copied to other diskettes, and then be VALIDATED as soon as possible.

Command Name: VALIDATE

Abbreviation: V

Purpose: This command cleans up the Block Availability Map by recreating it through tracing each file's starting and successive track and sector links. This process cannot follow the block or sector allocation schemes used by random or relative files and should not be used on any diskettes containing relative or random files.

Syntax: PRINT# fileno, "VALIDATE driveno(optional)"

Example: PRINT#15, "V"  
PRINT#15, "VALIDATE0"

Arguments: PRINT# -- prints the following command to the previously opened file or device number.

fileno -- refers to the previously opened file number. The original file number should have been any number between 1 and 127 that was not currently open. This file should have been opened using the disk device's command channel number (15).

drvno -- the drive number, 0 or 1, within each device number.

**Problem Preventor:** Due to its disregard for random or relative file types this command should be used with extreme caution. If the command is performed on diskettes containing any relative files, those files may be permanently lost.

One of the best arguments for using the VALIDATE command has to do with recovering from unclosed files. A file whose file type is preceded with an asterisk (\*) in a directory listing is an unclosed file. The VALIDATE process will erase or totally free the Block Availability Map (or BAM), then rebuild it from the track/sector links of the directory entries. An unclosed file is changed to a 00 (Deleted) status, and is not traced for block or sector usage during this process. As a result the offending unclosed file will no longer pose a threat to the integrity of your diskette filing system.

The VALIDATE process rebuilds the BAM image in its own local RAM until the entire list of directory entries has been successfully traced through its track/sector links. As a result, any error conditions encountered in its process of rebuilding the BAM will not alter the real data on your diskette. Only when a successful rebuilding of the BAM has occurred does the old image get replaced with the new one.

#### Advanced Disk Commands

These commands have to do with some advanced programming capabilities of your disk drive and are not to be attempted lightly even by experienced programmers. The structure and syntax of these commands are presented in outline form here for your general awareness; any elaboration on these commands if felt to be beyond the scope of this document.

| Command<br>----- | Syntax<br>-----   | Usage<br>-----  |
|------------------|---|---|
| BLOCK-READ       | PRINT# fil#, "B-R"; ch#; tk; sc   | Moves a given track/sector to a buffer location in the drive's RAM                            |
| BUFFER-POINTER   | PRINT# fil#, "B-P"; ch#; bytpos   | Allows access to any individual byte in the drive's RAM buffers                               |
| BLOCK-WRITE      | PRINT# fil#, "B-W"; ch#; drf#; tk; sc                                       | Allows the drive's RAM buffer to be written to any track and sector on the disk               |
| MEMORY-READ      | PRINT# fil#, "M-R" CHR\$(lo-byt)<br>CHR\$(hi-byt) CHR\$(# of bytes)         | Allows any portion of the drive's RAM, ROM or control addresses to be read into the computer. |
| MEMORY-WRITE     | PRINT# fil#, "M-W" CHR\$(lo-byt)<br>CHR\$(hi-byt) CHR\$(# of bytes)<br>DATA | Allows a block of data to be written into the drive's RAM buffer                              |
| BLOCK-ALLOCATE   | PRINT# fil#, "B-A"; dr#; tk; sc   | Allows you to directly designate a sector in BAM as being in use                              |
| BLOCK-FREE       | PRINT# fil#, "B-F"; dr#; tk; sc   | Allows you to directly designate a sector in BAM as being free                                |
| MEMORY-EXECUTE   | PRINT# fil#, "M-E" CHR\$(lo-byt)<br>CHR\$(hi-byt)                           | Allows you to execute ROM or RAM routines which are resident in the drive                     |
| BLOCK-EXECUTE    | PRINT# fil#, "B-E"; ch#; dr#; tk; sc  | Loads a sector from disk into one of the drive's RAM buffers & executes it                    |

| Command  | Syntax   | Usage  |
|----------|--|--|
|          | -----  | -----  |
| POSITION | PRINT# fi#, "P" CHR\$(ch#+96)<br>CHR\$(rec-lo) CHR\$(rec-hi)<br>CHR\$(rec-pos) | Positionsthe file pointer<br>to the correct record and<br>position within the record<br>for RELative files |
| UI+      | PRINT# fi#, "UI+"  | Set drive to Commodore 64<br>communications speed  |
| UI-      | PRINT# fi#, "UI-"  | Set drive to VIC 20 com-<br>munications speed  |

Arguments for Advanced Disk Commands

PRINT#-- prints the following command to the previously opened file, channel or device number.  
 fi# -- refers to the previously opened logical file number. The original file number should have been any number between 1 and 127 that was not currently open.  
 dr# -- the drive number, 0 or 1, within each device number.  
 ch# -- the secondary address (Channel#) of the associated OPEN command.  
 tk -- track, numbering 1 to 35  
 sc -- sector, numbering 0 to the maximum range for that track  
 bytes -- a number from 0 to 255 indicating which byte within the 256 byte buffer  
 lo-byt -- the low byte of a full 16-bit address  
 hi-byt -- the high byte of a full 16-bit address  
 # of bytes -- the number of bytes (1 - 255) to be sent from or to the drive's memory  
 rec-lo -- the low byte of a 16-bit record number  
 rec-hi -- the high byte of a 16-bit record number  
 rec-pos -- a number from 0 to 255 indicating which byte within the 256 byte record

#### USING THE GT "ROM DRIVE"

In order to allow rapid and convenient access to some of the GT Disk Drive's most powerful and useful features, several utility programs have been permanently stored in a "ROM Drive" within the normal disk drive.

You can reach the "ROM Drive" by loading from, or asking for a directory of drive number 1 on any Indus GT. For example, requesting LOAD "\$1",8 and then typing LIST will yield the following directory (provided device 8 is an Indus GT):

```

1 "INDUS GT C64      " ID 2A
1   "FIO & DW"          PRG
1   "FIO"              PRG
2   "FAST I/O"          PRG
1   "DW"               PRG
3   "DOS WEDGE"        PRG
1   "FC"               PRG
5   "FAST COPY"         PRG
0 BLOCKS FREE

```

NOTE: If your directory is slightly different, your Indus GT is probably newer than this manual. (Check for an Addendum or Errata sheet for any recent updates.)

Each of these utility programs will be discussed in some detail below. Several of the differences between a "ROM Drive" and a normal disk drive should be noted at this point:

- 1. The "ROM Drive" has been provided as a read-only convenience for several powerful utility programs. It cannot be SAVED to, or SCRATCHed, or VALIDATED, or any other DOS functions other than LOAD and "\$1" (Directory).
- 2. Saves and other attempts to alter data on the "ROM Drive" will yield WRITE PROTECT ERROR responses.

3. Identifying the GT Disk Drive to your system or application software as anything other than a single disk drive can cause unpredictable results and should be avoided at all times. The "ROM Drive" should be used only to initially load any of the utility programs you wish to use and should then be ignored entirely.

**NOTE:** Indus treats the contents of the "ROM Drive" just like any other diskette distributed with the Indus GT and reserves the right to make improvements or changes in its contents at any time and without notice.

**PROBLEM PREVENTOR:** Each of the functions performed by the utilities included on the "ROM Drive" make use of very special functions within the Disk Operating System (DOS) contained in the drive. Many copy protected disks avoid performing certain tasks using the normal DOS methods, and will therefore not be able to take advantage of the improved performance or ease of use these programs provide. You will need to experiment with any programs you wish to use in order to ascertain if any incompatibilities exist with the enhanced performance capabilities. Try starting with the combined loader (LOAD "1:","",8,1), and if any incompatibilities exist, try separately loading each of the functions (LOAD "1:DW",8,1 for the DOS WEDGE; or LOAD "1:FIO",8,1 for the FAST I/O) to individually test for compatibility. Also, you might get into the habit of issuing some "dummy" command on drive 0 (the "real" drive) after loading any utilities from the "ROM drive" (drive 1). For instance, getting a directory from drive 0 (using "#\$0" after doing a LOAD "1:","",8,1) could be a good practice.

**PROBLEM PREVENTOR:** Since the existence of a drive 1 (albeit a "ROM Drive") has been invoked inside our single drive unit, it will pay to specify a "0:" prior to your file name more often than you might normally have done. This will eliminate or lessen the chances of the DOS continuing to reference the last drive (which might have been our "ROM Drive" "1:") instead of switching to its normal default of Drive 0:.

Each of the Utility programs residing on the "ROM Drive" is preceded by a loader program (essentially a binary type loader) that is to be addressed for loading purposes with a ",1" following the Device number specified. These loader programs consist of acronyms of their functions (e.g., "FC" for "FAST COPY").

#### FAST I/O (FIO)

This program, invoked by LOADING "1:FIO",8,1 (or % 1:FIO when using the DOS Wedge). Once engaged it will load and save programs when using normal DOS LOAD and SAVE command parameters much quicker than under normal DOS control by actually sending information over the serial channel several times faster than usual. However, many copy-protected programs avoid using normal DOS functions when loading from and saving to their diskettes and will not be able to take advantage of the improved performance this program affords. Since the drive will be ready to load the next sector from the disk sooner than under normal DOS loading, additional loading speed can be achieved when the drive's files have been previously saved using the FAST I/O's save routines (the sector interleaving will have been optimized to lessen the latency or time spent waiting for the next sector to come around as the diskette is spinning).

If some heavily protected program causes the FAST I/O routine to stop operating, the FAST I/O can normally be re-started by typing SYS 51200 and pressing the RETURN key. But if this doesn't work, you can always reload it from the drive.

## DOS WEDGE (DW)

The DOS Wedge program, which can be started by LOADING "1:DW",8,1, is a very versatile and powerful DOS management aid. Many hours of keystroking can be eliminated when this program has been installed. A brief comparison of the DOS Wedge's command shorthand with the normal DOS methods of accomplishing the same objectives follows. In addition, the DOS Wedge allows you to get diskette directories without losing any program you have in memory. The advantages to a serious programmer are immense!

| FUNCTION  | DOS WEDGE<br>COMMAND | NORMAL DOS<br>COMMAND    | WHAT THE COMMAND DOES  |
|---|----------------------|--------------------------|--|
| LOAD  | /filename            | LOAD "filename",8        | Loads a file from the disk into the computer's memory.   |
| LOAD & RUN  | !filename            | LOAD "filename",8<br>RUN | Loads a file from the disk into the computer's memory then runs or executes it.  |
| LOAD & RUN<br>to address<br>(Autoload,<br>same as ,1) | %filename            | LOAD "filename",8,1      | Loads a file from disk directly to a location in the computer's memory, then runs or executes (All of the " ROM Drive " Utilities operate with a % command). |
| SAVE  | -filename            | SAVE "filename",8        | Copies a file from the computer's memory to the disk. Also records data about the file on the disk's directory.  |
| SAVE &<br>REPLACE                                     | #0:filename          | SAVE "#0:filename",8     | Writes a file in the computer's memory over an existing file on the disk. Updates the disk's directory with the new file's data.                             |
| DIRECTORY   | \$                   | LOAD "\$",8<br>LIST      | Loads a list of file names from the diskette and displays them   |

| FUNCTION            | DOS WEDGE<br>COMMAND | NORMAL DOS<br>COMMAND                                | WHAT THE COMMAND DOES   |
|---------------------|----------------------|--|---|
|                     |                      |  | on the screen. The correct syntax for viewing a directory of the "ROM Drive" is to LOAD "\$1,B then LIST ( or # \$: when the DOS Wedge has been engaged). This command will not interfere with any program you have in memory.  |
| RENAME              | #R0:new=0:old        | OPEN 15,8,15<br>PRINT#15,"#R0:new=0:old"<br>CLOSE 15 | Permits a filename on the disk to be changed.   |
| NEW<br>(format)     | #NO:name,id          | OPEN 15,8,15<br>PRINT#15,"#NO:name,id"<br>CLOSE 15   | Places track, sector, and directory information on a blank diskette; erases and reformats information on old diskettes. If you do not provide an id number, the old data will be erased but the disk track and sector blocks (format) will remain. <u>Be careful!</u> |
| SCRATCH<br>(delete) | #S0:filename         | OPEN 15,8,15<br>PRINT#15,"#S0:filename"<br>CLOSE 15  | Deletes or erases a file from the disk, freeing the space it occupied for other programs.   |
| COPY                | #CO:new=0:old        | OPEN 15,8,15<br>PRINT#15,"#CO:new=0:old"<br>CLOSE 15 | Loads a file from the disk and resaves it to the disk using a new name.   |
| INITIALIZE #IO      |                      | OPEN 15,8,15<br>PRINT#15,"#IO"<br>CLOSE 15           | Should be used whenever a new diskette is inserted in the drive or when the system is first powered up. Forces the  |

| FUNCTION            | DOS WEDGE COMMAND | NORMAL DOS COMMAND   | WHAT THE COMMAND DOES   |
|---------------------|-------------------|--|---|
|                     |                   |  | drive to read the disk id and Block Allocation Map from the diskette and store it internally. Also used to clear flashing error light.  |
| VALIDATE            | EVO               | OPEN 15,8,15<br>PRINT#15,"V0"<br>CLOSE 15  | First de-allocates all of the blocks in the BAM, then traces through all files in the directory to re-build the BAM from the track/sector links in each file. Should not be used with relative files. |
| ERROR DISPLAY       | E                 | 10 OPEN 15,8,15<br>20 INPUT#15,A\$,B\$,C\$,D\$condition causing the disk<br>30 PRINT A\$,B\$,C\$,D\$drive's busy light to flash. | Displays on the screen the error  |
| SCREEN COLOR CHANGE | E                 | POKE 53280,color<br>POKE 53281,color   | Changes background and text colors for better viewing on your TV or monitor. The DOS Wedge version cycles through each of the 16 colors each time its symbol is pressed and a RETURN key is hit.      |
| RESET DRIVE#        | #@newdev          | none required  | Resets the default drive number for DOS Wedge functions.  |
| DISABLE WEDGE       | EQ                | none required  | Exits the DOS Wedge utility.  |
| ENABLE WEDGE        | SIS 52224         | none required  | Re-enters the DOS Wedge once it has been previously loaded.   |

As with other memory resident programs and the other utilities provided with your GT Disk Drive, some copy protected programs will have problems with the DOS Wedge performing some of its functions. If this is the case with your program, simply use the EQ command and try the same function with the normal DOS long hand method.

#### FAST COPY (FC)

The fast copy routine is entered by LOADING "1:FC",8,1 (or \$ 1:FC when the DOS Wedge has been engaged). When loaded you will be prompted to "INSERT SOURCE DISK AND PRESS RETURN." When you have inserted a source disk and hit RETURN, you will be shown "READING TRACK XX SECTOR XX". When it has loaded as much as it can fit into the computer's memory, you will be prompted to "INSERT TARGET DISK AND PRESS RETURN", and so on until the entire diskette is copied. It will take about 4 diskette interchanges to completely copy your diskette, but since this copy is extra fast only a fraction of the normal time required to copy a diskette will be used. Needless to say, copy protected diskettes are protected from being copied by programs such as these, and will not work.

## USING THE GT UTILITY DISKETTE

The GT Utility Disk contains a menu driven program that features several useful disk maintenance functions. You can take advantage of these functions by loading the GT Utility Disk into your disk drive and typing: LOAD "GTHELP", 8 ;and when the computer has responded with READY typing: RUN.

Before viewing the Main Menu of the program, you will be asked if you have a 1525 printer or not (the program will be able to determine if you actually have a printer attached but can't see if it is a 1525 variety or not). Answering Yes will set a flag in the program, enabling the graphic characters displayed on the screen during track/sector editing, etc., to be printed directly on your paper. If you do have a printer attached, but not a 1525 type, the graphic characters will be represented by a period (.) on your paper.

When the program has loaded, you will be greeted with the Menu Screen which contains the following information and choices:

```
GT UTILITIES  
BY MIKE LOUDER  
(C) 1983 DATAMOST INC.  
  
(C)OPYMASTER          (R)ENAME FILES  
(D)IRECTORY          (S)CRATCH FILES  
(E)XIT TO BASIC      (T)RACK/SECTOR  
(F)ORMAT DISK         (U)NDELETE FILES  
(N)EW DISK SET        (V)ALIDATE DISK  
  
ENTER COMMAND: _  
(WRITE-PROTECT IMPORTANT DISKS.)
```

The flashing underline cursor next to the "ENTER COMMAND:" statement, is requesting that you enter the first letter of one of the menu choices to get

to the next step in the program. In virtually every section of this program, hitting the RUN/STOP key will return you to this Main Menu screen; this can often come in handy, since not all of the sub-menu screens have their own exit commands available at each level. Each of these choices will be discussed below in the order that they appear on the screen.

Another feature available in virtually every section of this program is the toggling between character set 1 and 2 by using the SHIFT-COMMODORE command. This is particularly useful when viewing some of the track and sector data with the editor provided on this utility diskette, but can also reveal the contents of some programs and wordprocessors whose output has made use of the alternate character set provided by the Commodore computer.

### (C)OPYMASTER

Pressing C for Copymaster will get you into the disk and file copying utility. The next screen you see will give you another set of menu choices:

- (1) COPY FILES WITH PROMPTS
- (2) COPY FILES WITHOUT PROMPTS
- (3) COPY ENTIRE DISKETTE
- (Q)UIT

SELECT: \_

Pressing 1 will generate an "ENTER FILE NAME: " \_ response. Entering any valid file name or wild card character at this point will generate the screen shown below:

```
READING: T-18 S-01  
IN DEVICE 8  
ENTER FILE NAME: **  
"GTHELP.BOOTER"          (Y)ES/(N)O/(Q)UIT? _
```

Answering Yes or No will result in the appropriate response being made (the file being copied or not, as you have chosen) and your choice being noted on the screen. You will then be prompted for each file on the directory matching

your file name specifications, until no more directory entries are found at which time you will be asked: "FILE NOT FOUND. CONTINUE? (Y)ES/(N)O ". You can continue looking for your file name on other diskettes or quit.

The 2 choice will copy active sequential or program files, but will only prompt for inserting target and source diskettes as required, then asking you to type G for Go when ready.

The 3 choice is the full disk copy choice, and should be used whenever relative or random files are to be copied. You will be instructed to insert the source and target diskettes as required and will be shown a view of the Block Availability Map which graphically depicts the "active" sectors (sectors with data in them) as they are being read and written.

#### (D)IRECTORY

Pressing D for directory will give you a list of programs on the diskette. However, this command gives you much more information than LOADING and LISTING a \$, as we will see below. The Directory command is the first time you will notice that your GT Utilities program has noticed that you have a printer plugged in and turned on (if you do have one). Those having active, powered on printers, will next see a screen asking, "PRINT DIRECTORY? (Y/N): ". Answering Y or N to this question will advance you to the next question; once the next question is answered, the directory listing will be made to the printer and to the screen if you answered Y or only to the screen if you answered N to the previous question. If you do not have a printer powered on and connected to your computer, you will not be asked this question.

The next question posed demonstrates one of the differences in the directory program on this utility diskette: "START ADDRESS (H)EX//D)BC//(N)ONE ". In addition to printing a list of the files contained on the disk, this directory program will inform you of the load address of each of the program files. It will give this information in hexadecimal form, decimal form, or will print a list of the files without this information (the "None" choice). If you are not looking for this load address information, the directory listing will occur much faster if you press the N for None choice (the program will then not have to go the beginning track and sector of each directory entry to check for this initial load address information). Pressing H, D, or N will cause your printer to begin listing the directory (if you have a printer, and if you answered Y to the previous question). In any case you will now have displayed

on your screen the following information:

#### DIRECTORY

READING:T-17 S-01

IN DEVICE 8

#### "Title of Diskette"

| BLKS | TITLE        | TYPE  | TK | SC | ADDR   |
|------|--------------|-------|----|----|--------|
| 2    | "GTHELP      | " PRG | 17 | 00 | \$0801 |
| 43   | "GTHELP.CODE | " PRG | 17 | 01 | \$1000 |

619 BLOCKS FREE

PRESS A KEY TO CONTINUE.

Each of the elements of this screen will be described briefly. The function (DIRECTORY) being performed is displayed in the upper-left hand portion of the screen. This function description is common to most of the Main Menu choices.

The upper right-hand corner of the screen shows reading/writing sources or destinations in tracks and sectors, as well as the device numbers involved.

The diskette's title and id number are displayed next. Below the boxed off area is where the bulk of the information is contained. The first section, under the heading "BLKS", displays the number of blocks or sectors the file occupies (each sector or block contains 256 bytes of information). Under the "TITLE" heading you will find the file's title or name. Since this Title column employs a special "quote" mode to display its information, directory entries containing special characters not normally visible with LOADING and LISTING "#". Under the "TYPE" heading you will find the PRG, SEQ, etc. The first three headings of the directory yield the same information as LOADING and LISTING a "\$" from the diskette, with the exception that instead of omitting a DELETED file, a "---" will be shown under the "TYPE" heading. This is a very important piece of information, since any files showing up under the directory with a "---" TYPE can be potentially recovered using the "UNDELETE" command described later in this manual.

The next two headings "TK" and "SC", contain the track and sector numbers where each particular file begins. If you watched closely when you responded to the "START ADDRESS" prompt, the directory program first looked for the

files on track 18, sector 00, then moved to the starting track and sector of each file to look up the initial load address of each file. The Track and Sector locations in the upper-right hand corner of the screen will probably match the initial track and sector location of the last file on the screen.

Under the "ADDR" heading you will find the locations in hexadecimal or decimal (if you pressed H or D earlier, or nothing if you pressed N) of the initial load address of a program file. This information saves you loading the starting track and sector of the file and inspecting the third and fourth bytes of the sector (lo-byte and hi-byte) for the load address of the program. Knowing this information can prove very useful at times. For instance, load address 2049 decimal (or hex \$0801) is where basic programs are typically stored; a file loading at decimal 53248 (or hex \$D000) would probably contain sprite characters, etc.

You will continue to be prompted "PRESS A KEY TO CONTINUE. " until the directory program has shown all of the files on the diskette (a diskette can contain up to 144 directory entries). When all of the files have been shown, a message of "XXX BLOCKS FREE" will be given, and hitting any key will return you to the Main Menu. Remember, if you found the directory entry you need before seeing the rest of the directory screen entries, you can exit the directory function by hitting the RUN/STOP key.

#### (E)XIT TO BASIC

Pressing the E key from the main menu will cause you to quit the GT Utility program and return to BASIC. Although you will have left the program itself, the loader portion of the GT Utility program will probably still be resident, and typing SYS 4096 will restore the main menu screen for you. Since the GT Utilities program turns off the FAST I/O function during the course of performing its normal duties, you will be shown that it has been restored (if it was previously loaded) during the Exit process.

#### (F)ORMAT DISK

Pressing F will display the "FORMAT" function in the upper left screen, and ask for "FILE NAME ONLY: \_". You can enter a name of up to 16 characters in length (normally only alpha-numeric characters should be used). As with the

"NEW" command this diskette name is for cosmetic or reference purposes only. Since this function is performing essentially the same chore as the disk "NEW" command, the next request for "# CHARACTER ID: " has similar significance (that is if no id number is provided, the old files are erased but the diskette does not re-format the diskettes into tracks and sectors). PLEASE NOTE: Using this command will destroy any information that exists on your diskettes; make certain you know what diskette you are formatting before you begin -- the UNDELETE utility will not be able to recover a file lost by reformatting your diskette.

#### (N)EW DISK SET

Pressing the N will generate a screen showing "RESET SOURCE AND TARGET DISK DRIVE". Under this the current source and target disk drives will be shown; typically: "SOURCE DEVICE: 8 TARGET DEVICE: 8". If you wish to return to the Main Menu at this point, you can do so by pressing M (or RUN/STOP). Otherwise you are prompted for "INPUT SOURCE DRIVE DEVICE NUMBER \_", and then "INPUT TARGET DRIVE DEVICE NUMBER \_". This program is limited to sourcing and targeting only two devices, either 8 or 9. You will not be able to fool the program by specifying a device number that is not present; the NEW DISK SET utility will inform you the device is not present and return to the previous default disk number (the device number from which the GT Utility program was loaded). Resetting the source and target device numbers can be particularly useful when using the COPYMASTER and UNDELETE functions of this diskette.

#### (R)ENAME FILES

Engaging the Rename Files function by pressing R will prompt first, "NEW FILE NAME \_" (up to 16 characters) and then "OLD FILE NAME \_". Errors involving files not found, etc., can be minimized by checking a directory of the diskette for the proper name of the file you wish to rename.

#### (S)CRATCH FILES

The SCRATCH or delete files command performs essentially the same function as the direct disk command SCRATCH, with the exception that if you press an "\*" in response to the "FILE NAME \_" prompt, you will be given a second chance.

The SCRATCH FILES utility will ask, "DO YOU WANT TO SCRATCH ALL FILES? \_" before continuing. A Y will delete all files on the diskette, a N will return you to the beginning of the SCRATCH FILES program.

#### (T)RACK/SECTOR

The TRACK/SECTOR editor is an invaluable tool for those wanting to learn more about what is going on in their disk drives and computers. Pressing T will bring up the following screen:

```
(R)EAD   (W)RITE   (M)AP   (P)AGE   SELECT: _

(T)RACK-XX   (S)ECTOR-XX   (C)HANGE BYTES
```

Pressing R for READ will cause the TRACK/SECTOR program to read the track and sector number noted next to their respective commands on the screen (unless you change them, this is usually the start of the Block Availability Map on Track 18, Sector 00). When R (or any of the other choices) is made, the upper left hand portion of the screen momentarily informs you of the Track, Sector, and Device number being affected, then returns to the "SELECT: \_" prompt. By pressing T for Track, you will be prompted for a two digit track number, then be immediately switched to being prompted for a two digit sector number (without pressing S for Sector). If you did not press T for Track, pressing only the S for Sector would let you choose to view a different sector on the same track.

The W for Write choice will record the sector currently being viewed (including any changes you have made) back to the same track and sector number from which it came. The Write function will store a new CRC or checksum character for any changed data fields, if required, and will do so without asking. (You don't get a chance to intentionally or unintentionally write an illegal CRC number.)

Pressing the M for Map will present you with a graphic representation of the Block Availability Map. The TRACK/SECTOR program will automatically set the track and sector to Track 18 Sector 00 when you press the M. Each sector containing data will be shown by a "\*" appearing on the track/sector grid; sectors without data are shown as blank spaces. In the upper right corner of the screen, you will be prompted to press G for Go to return to the Main Menu.

If you have a printer connected and powered on you will also be prompted to press P for Printer if you want a hard-copy output of the screen. (If you do not have a printer, this message will not appear.) Knowing which tracks and sectors contain data can save hours of time searching the disk for the information you desire.

When the Map display is shown, the entire set of tracks and sectors are on the screen at one time. All other Track and Sector displays have room for only the first 128 bytes (in Hexadecimal) to be shown on the screen at one time. The second 128 bytes can be viewed by pressing the P for Page command. The beginning byte numbers of each line are shown on the leftmost side of the screen (they are of course numbered hexidecially since they are counting hexadecimal bytes). On the rightmost side of the screen is the screen display code equivalent of the hex characters shown on the left. The screen display codes normally shown are the primary character set (Set 1). Pressing the SHIFT-COMMODORE keys can be immensely helpful in decrypting information written in some sectors (a graphic heart symbol will become a more intelligible capital S), by letting you switch to the alternate character set (Set 2). Pressing the SHIFT-COMMODORE again will return you to character set 1 again (another toggle switch!).

Since the P command character was utilized for Paging, printing the contents of a Sector you have Read or Changed is accomplished (of course, only if you have a printer) by pressing SHIFT-?. Although the right side of the screen always displays the primary character set until the SHIFT-COMMODORE keys are pressed, the output to your printer will display the character set (either 1 or 2) that the program intended to write.

The C for Change Bytes function, does exactly what you'd expect. You can use your CURSOR-UP/DOWN and CURSOR-LEFT/RIGHT keys to edit the hexadecimal bytes on the left side of the screen. Each byte changed will show up as a new character in the screen display code on the right side of the screen. When you have changed all you want to change (including the second page of your sector), pressing Q for Quitting the Change command will allow you to Write your changed sector back to the diskette or go on to some other function.

The bottom of the screen offers additional choices for the top of the screen's "SELECT: \_" prompt. The first two bytes of each sector (bytes \$00 and \$01 -- not bytes \$80 and \$81 on the second page of the sector) contain the pointer to the next track and sector of this file. Byte 1 is the hexadecimal value of

the next track, byte 2 is the hexidecimal value of the next sector. To assist you in following a file through all of its related sectors, the TRACK/SECTOR program looks at these bytes and will take you to the next sector and track of this file by pressing N for Next. If a 00 or some other illegal track or sector numbers appear in bytes 1 or 2, instead of being prompted for (N)EXT track and sector, you will be informed, "END OF TRACK/SECTOR LINK." The other Select choice given at the bottom of the screen is a Q for Quit, which can be used along with RUN/STOP to return you to the Main Menu.

#### (U)UNDELETE FILES

The U for Undelete choice is one of the most helpful commands for beginners (and for some embarrassed advanced programmers) because it gives you the potential for being able to recover files which you have inadvertently deleted.

Some introductory precautions should be noted here:

Your chances of recovering a deleted file are greatest if the recovery is attempted immediately after the unintended deletion or "scratch" has occurred (immediately, meaning that no successive SAVEing or writing has been done to the diskette). Once a file has been deleted, the Block Availability Map allows any other program to use those sectors. You might get lucky, and a successive write to the disk not use one of the sectors your deleted file previously occupied, but don't count on it.

The Directory command on the GT Utility diskette will list all files on the diskette that stand a chance of being recovered. As previously noted these files contain three dashes (---) in their file TYPE columns. If the file does not appear on this directory, you will not be able to recover or UNDELETE it.

If there is a limited number of free blocks, if the file to be recovered is very large and/or if more than one deleted file is to be recovered, exceptional care should be exercised in recovering the affected files. It would best to have another pre-formatted diskette available on which to place the reconstructed or UNDELETED file. The recovered files can later be copied back to their original sources if so desired.

Files that have been physically damaged, or have become magnetically altered, or have mismatching data/CRC or track-sector/CRC mis-matches, will not be recoverable with this utility.

The UNDELETE choice will prompt you to "ENTER FILE NAME: \_". Using the title of a deleted file from your directory or an asterisk or other wild card can appear here. When a matching file name has been located, it will be displayed in quotes, and you will be asked "(Y)ES/(N)O/(Q)UIT? \_". If you respond Y, the UNDELETE program will begin tracing down all of the sectors related to this file, beginning with the track/sector link contained in bytes \$00 and \$01 of the beginning track and sector of the file (shown on the directory listing), and continuing through all of the sectors belonging to the file until the end of file (no legal forward track and sector number) is found.

You will next be given the choice of inserting a new diskette on which to save the UNDELETED file or leaving the same diskette in the drive when you are requested to: "PLACE TARGET DISK IN DEVICE 8. (G)O". Press G for Go when you have the diskette you want to use in the drive. The program will inform you that it is, "SAVING \"filename\"", as it writes the information to the diskette. Before finishing the reconstruction of your file, you will be asked to "ENTER FILE TYPE: (SEQ OR (P)RG: \_)". When you have informed the UNDELETE utility of the file type, it will note the information in the file's directory entry and inform you, "\"filename\" COPY COMPLETE", and go on to the next file matching your filename entry, or inform you, "FILE NOT FOUND. CONTINUE? (Y)ES/(N)O". An N or a RUN/STOP will now return you to the main menu.

#### (V)VALIDATE

This choice performs essentially the same function as the direct DOS command VALIDATE, without all of the typing and other trouble. The first prompt re-emphasizes the primary short-coming of the Validate process, that of not being able to handle relative or random files. You are asked, "DOES THIS DISK CONTAIN RANDOM FILES? (YES=ABORT TO MAIN MENU, NO=OK) \_". When you say No, you will be informed that the program is going about it's validating process and to please wait. A successful completion will be so noted, and you can go back to the Main Menu.

#### DISK DRIVE ERROR CODES

These error codes are generated by the disk drive, not usually as a fault of, or an error in, the drive, but rather as a means of helping you identify problems with the media or data when the disk drive encounters difficulty in reading data or writing to faulty media. What follows is a full list of the disk drive error codes which might be displayed when an error has occurred and the ERROR pushbutton on the drive's operator control panel has been pressed.

| ERROR CODE | ERROR MESSAGE  |
|------------|--|
| 00         | <u>No error</u>  |
| 01         | <u>Scratch File Response</u> - No error. This is the standard code given by the disk drive after a "SCRATCH FILE" command.   |
| 20         | <u>Block Header Not Found</u> - Read Error. The disk drive was unable to locate the block header for the block that was requested.   |
| 21         | <u>Sync Mark Not Found</u> - Read Error. The disk drive was unable to locate a SYNC mark on the track that was requested.  |
| 22         | <u>Data Block Not Found</u> - Read Error. The disk drive was unable to locate the requested data block.  |
| 23         | <u>Data Block Checksum Error</u> - Read Error. The data block that was read contained one or more bad bits, which caused the calculated checksum to not match the checksum that was written with the data. |
| 24         | <u>Byte Decode Error</u> - Read Error. An invalid bit pattern existed in the byte that was just read.  |
| 25         | <u>Bad Write Verify</u> - Write Error. Data that was read from the disk does not agree with data that was written to the disk.   |

| ERROR CODE | ERROR MESSAGE  |
|------------|--|
| 26         | <u>Write Protected</u> - An attempt was made to write to the disk while the diskette or the drive was write protected.   |
| 27         | <u>Header Checksum Error</u> - Read Error. The header that was read contained one or more bad bits, which caused the calculated checksum to not match the checksum that was written with the header. |
| 28         | <u>Data Block Overrun</u> - Write Error. The data block that was just written over-wrote the SYNC mark of the next block header.   |
| 29         | <u>Disk ID Mismatch</u> - The diskette in the drive was changed without notifying the drive.   |
| 30         | → <u>Syntax Error</u> - The drive recognized the command that was given, but there was an error in the number or type of parameters passed for the command.  |
| 31         | → <u>Syntax Error</u> - The drive did not recognize the command that was given.  |
| 32         | <u>Syntax Error</u> - The command line given contained too many characters.  |
| 33         | <u>Syntax Error</u> - An invalid file name was used in the command line.   |
| 34         | <u>Syntax Error</u> - The file name was missing from the command line or not recognized as a file name.  |
| 39         | <u>Syntax Error</u> - Invalid command on secondary command channel.  |
| 50         | <u>Record Not Found</u> - The record number requested during a GET# or INPUT# was higher than the last record number of the relative file.   |

| ERROR CODE | ERROR MESSAGE   |
|------------|---|
| 51         | <u>Record Size Overflow</u> - Data written to the record exceeded the definition of the record size.  |
| 52         | <u>File Too Large</u> - The current position of the record within a relative file will cause a disk overflow.   |
| 60         | <u>Write File Open</u> - An attempt was made to OPEN a file for reading that was still OPEN after a write.  |
| 61         | <u>File Not Open</u> - An attempt was made to read or write to a file that was not first OPENed.  |
| 62         | <u>File Not Found</u> - The file requested does not exist on the current diskette.  |
| 63         | <u>File Exists</u> - An attempt was made to create a file with a file name that was already present on the current diskette.                          |
| 64         | <u>File Type Mismatch</u> - The file type found in the directory does not match the file type that was requested.                                     |
| 65         | <u>No Block</u> - An attempt was made to allocate a block that was previously allocated.  |
| 66         | <u>Illegal Track or Sector</u> - An attempt was made to read from an invalid track or sector number.  |
| 67         | <u>Illegal System Track or Sector</u> - An attempt was made by the drive to read a track or sector that was not supported on the current disk format. |
| 70         | <u>No Channel Available</u> - The requested communications channel was previously allocated.  |
| 71         | <u>Directory Error</u> - The Block Allocation Map that the drive  |

| ERROR CODE | ERROR MESSAGE  |
|------------|--|
|            | is keeping does not match the Block Allocation Map on the diskette.  |
| 72         | <u>Disk Full</u> - The disk or directory is full. The diskette is limited to 144 directory entries and between 167,132 and 168,656 data bytes per diskette (depending on the types of records being written - relative files, sequential files, etc). The "Disk Full" error indicates that one or more of these parameters has been exceeded.              |
| 73         | <u>Dos Mismatch Error</u> - The DOS built into your Indus GT Disk Drive and into currently manufactured Commodore 1541 disk drives is not compatible with older DOS versions (DOS 1) manufactured by Commodore. Utilities are available that can be used to update programs requiring the older DOS versions; check with your computer dealer for details. |
| 74         | <u>Drive Not Ready</u> - This error is usually the result of sending some command or requesting status of a drive that does not contain a diskette. It can also result from some other hardware failure in the drive.  |

#### CARE AND USE OF YOUR DISKETTES

The magnetic surface of a floppy disk is a somewhat similar to that of a cassette tape. Each of the diskettes consists of a plastic base (normally mylar) coated with a magnetic material that your data can be stored on and read back from at a later time. The primary difference is that the floppy disk is enclosed in a flexible black plastic envelope rather than a hard plastic case, and that the floppy disk is designed for quick, random access similar to the way you can quickly select a particular song when using a phonograph record versus the serial selection process you must go through when using cassette tapes.

The precautions noted below concerning diskette care become obviously significant once it is realized that your valuable data is being written in a series of ones and zeros at a rate of more than 5,000 bits per inch on a material that heat and humidity can cause to expand and contract as much as several thousandths of an inch. Keeping track of the micro-volts of energy generated by these tiny flux changes passing over the drive's magnetic read/write head where timing of the bits being read is critical to billionths of a second, make paying strict attention to these safeguards a matter of high priority.

1. Store your diskettes away from direct sunlight. Keep them away from excessive heat. They can easily become warped. In normal operation, the ambient or room temperature should be between 50 and 122 degrees Fahrenheit (10 to 50 degrees Celsius).
2. Keep your diskettes away from excessive moisture and humidity. Diskettes are normally required to have between 8 and 80% relative humidity environments for best operation. Never wet or wash a diskette. The diskette envelope contains its own cleaning materials and lubricants.
3. Do not bend your diskettes, handle them with care when loading or unloading from your disk drive. They must be allowed to turn freely within their envelope.
4. Store your diskettes in their specially treated sleeves and keep them standing on edge to prevent damage to the magnetic surface.
5. Never touch the magnetic surface of the diskette where it is exposed

through the oval opening in the black plastic envelope. Fingerprints can damage the magnetic medium and render the surface unusable.

6. Never allow your diskettes to be stored within the reach of a magnetic field, such as your monitor, or the telephone when it rings. Magnetic fields can erase the data on your diskette.
7. Do not attach paper clips or staples to your diskettes.
8. Do not write on your diskettes with a ball-point pen or pencil. Use a felt-tip pen to mark on the diskette label, or write on the label before you put it on the diskette.
9. Do not use pencil erasers on diskette labels as the eraser dust is very abrasive and can damage the diskette.
10. Do use good common sense in preserving the life of your diskettes. Remember, the data you save may be your own.

#### **NOISE RADIATION NOTICE**

Section 15.838 of the FCC rules and regulations stipulates that the following information be provided to users of Class B computing devices regarding the interference potential of the devices and simple measures that can be taken by the user to correct the interference:

This equipment generates and uses radio frequency energy and if not installed and used properly, i.e., in strict accordance with the manufacturer's instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

Reorient the receiving antenna.

Relocate the computer with respect to the receiver.

Move the computer away from the receiver.

Plug the computer into a different branch outlet so that the computer and the receiver are on different branch circuits.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet prepared by the Federal Communications Commission helpful:

"How to Identify and Resolve Radio-TV Interference Problems"

This booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402. Stock No. 004-000-00345-4.

**IMPORTANT NOTE:** This product is supplied with a shielded serial interface cable. The use of a cable other than that supplied may defeat the RFI shielding of the communications between the computer and the disk drive and is therefore not recommended.

#### **NOTICE**

Indus System Inc. reserves the right to make improvements in the product described herein at any time and without notice.

#### **DISCLAIMER**

Indus Systems Inc. shall have no liability or responsibility to the purchaser or any other person or entity with respect to any liability, loss or damage caused or alleged to be caused directly or indirectly by this manual or its use, including but not limited to any interruption in service, loss of business and anticipatory profits or consequential damages resulting from the use of this product.